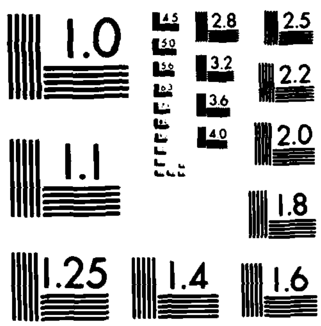
AD-A151 848 A DATABASE DESIGN FOR THE BRAZILIAN AIR FORCE FLYING 1/3
UNIT OPERATIONAL CON..(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. A M DA CUNHA
UNCLASSIFIED 14 DEC 84 AFIT/GCS/ENG/84D-7 F/G 9/2 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963 A

A DATABASE DESIGN
FOR THE BRAZILIAN AIR FORCE
FLYING UNIT OPERATIONAL CONTROL SYSTEM

THESIS

Adilson Marques da Cunha
Major, BRAZILIAN AIR FORCE

AFIT/GCS/ENG/84D-7

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GCS/ENG/84D-7

A DATABASE DESIGN
FOR THE BRAZILIAN AIR FORCE
FLYING UNIT OPERATIONAL CONTROL SYSTEM

THESIS

Adilson Marques da Cunha
Major, BRAZILIAN AIR FORCE

AFIT/GCS/ENG/84D-7

Approved for public release; distribuition unlimited.

AFIT/GCS/ENG/84D-7

A DATABASE DESIGN
THE
FOR BRAZILIAN AIR FORCE
FLYING UNIT OPERATIONAL CONTROL SYSTEM

THESIS

Presented to Faculty of the School of Engineering

of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirements for the Degree of

Master of Science in Computer Systems

Adilson Marques da Cunha, B.S.

Major, BRAZILIAN AIR FORCE

14 December 1984

Approved for public release; distribuition unlimited

Table of Contents

Page

Preface

The purpose of this thesis is to design and partially implement a database application for the Brazilian Air Force Flying Unit Operational Control System.

In designing and writing this research I have had a great deal of help from others. I am deeply indebted to my thesis advisors, Capt Stephen E. Cross and Capt Patricia K. Lawlis, for their continuing patience and assistance in times of need. I wish to thank the AFIT faculty members, Dr. Thomas C. Hartrum and Dr. Henry B. Potoczny, for their important and worthy refinements. I also wish to thank Colonel Aly I. El Deihi from the Egyptian Army and Captain Dale Pontiff from the U.S. Air Force, for their help and cooperation during the design and implementation phases of this thesis. A word of thanks is also owed to Dr. Paulo Dantas Cabral, my advisor in the Brazilian Air Force, from whom I have received much motivation, incentive, and direction. My personal thanks also to the Brazilian Air Force that gave me this precious opportunity to be the first Brazilian officer to take the Master's degree course at the AFIT School of Engineering. Finally, I wish to thank my wife, Solange, for her understanding and concern on those many nights when I was tied to my desk with work.

Adilson Marques da Cunha

List of Figures

Figures Page

List of Tables

Abstract

This thesis addresses a relational database design for a Brazilian Air Force Flying Unit Operational Control System. After defining the problem and specifying requirements, an overall system analysis was performed using Decision Support System Theory. A top-down planning for decision support systems and databases, and a functional analysis were performed to identify potential environmental database applications. Using a canonical approach, a file management system was mapped to a database conceptual model and afterwards to a database logical model.

A prototype Dialog Generator Management Software was implemented through menu driven programs, using the dBASE II DBMS version 2.1 running on a Z-80 microcomputer. The database partial implementation was performed using the INGRES DBMS version 7.10 running on a VAX 11/780, from which advanced queries were retrieved. Finally, an investigation of optimum query retrievals from databases is performed using Artificial Intelligence methods and techniques.

A DATABASE DESIGN
FOR THE BRAZILIAN AIR FORCE
FLYING UNIT OPERATIONAL CONTROL SYSTEM

I. Introduction

1.1 Background

In 1975, the Brazilian Air Force (BAF) began to recognize the importance of having a computer system to aid in operational decision making. The Command Staff decided to create a System Analysis Group to develop a computerized file management system for flying unit statistics and decision making. As a member of this group, the author of this thesis worked in nearly all phases of the project, from the problem definition to the final test and implementation phase (9:12,32).

Requests increased as more users recognized the advantages of this automated file management system. Consequently, the level of usage became extremely high. However, even now, each unit data management request must be almost individually programmed, requiring redundant efforts. A problem with this system is that it does not serve a large number of users in a automated and intelligent manner.

Basically, the problem must be solved because a faster, more efficient, and effective way to process and control flying unit data, in support of flying unit operational control, is required for the BAF. A further increase in user demands is expected and the existing system will be unable to support the additional workload because of design limitations. Thus, a more responsive and advanced system becomes necessary to reduce redundant and outdated information because the current system is approaching its maximum performance boundaries (4:19,22).

1.2 Problem Definition

The problem consists of increasing the efficiency of the flying unit operational decision making, which is currently implemented as a file management system in the Brazilian Air Force. The present system is approaching its maximum performance boundaries. In order to increase efficiency beyond current levels, an improved and integrated database application system must be designed. The main objective of this thesis effort is to design the database system and demonstrate its feasibility through a partial implementation.

1.3 Scope

The thesis deals only with information generated from military flights in executed missions (i.e., the information generated during the time interval between aircraft engine start-up and cut-off in the parking area). It encompasses the relationships between entities (aircraft, crew member, mission, etc.), judged important to perform efficient flying unit operating functions. Any other information out of this time interval is not covered because it does not belong to the scope of the project. Using methods of artificial intelligence, an investigation was performed as part of this project, to determine how intelligent retrieval from this type of database could be done. Mainly due to time constraints, only specific and selected types of information concerned with the operational control of an Air Force were considered.

1.4 Summary of Current Knowledge

Although there are several database application systems already available on the market, none of them addresses the specific characteristics of a flying unit operational control system. Current knowledge in the professional field related to this problem of flying units control systems for an Air Force environment is very controversial, nebulous and supported only by huge systems. Due to the classified nature of the research material, few bibliographic sources are available for research in this area. However, information and techniques are published on similar database using the artificial intelligence approach, which can be used to support this research project. Representative systems using natural language are discussed in Chapter VI.

1.5 General Approach and Standards

A top-down analysis approach was adopted to develop this thesis research using the Decision Support Systems (DSS) theory, database systems design techniques and artificial intelligence principles. Initially, four data groups were identified to gain an understanding of the problem during the analytical phase: personnel data, material data, operational data, and other specific data not included in the previous data groups. Since complex problems and systems are better understood when properly modularized and studied, this kind of approach seemed best, based upon the development of systems using similar characteristics and applications.

To design a database application, able to support useful selected relations for a flying unit operational control system, the following basic criteria were applied: simplicity, clarity, modularity, conciseness, and efficiency. Techniques such as modularization, divide and conquer, structured database design, and intelligent retrieval from databases were used to produce an improved solution. It was assumed that the final product of this self-contained database application would be initially installed in a mainframe computer environment. Afterwards, it is desirable that this system be adapted to work in microcomputer environments, because, if a portable flying unit operational control database system worked in a microcomputer environment, it could provide several tactical advantages for fast and dynamic decision making processes. It could, for instance, help to properly perform and measure "in locum" the control of flying units operations within a particular Squadron, Group, or even Air Force, during specific campaigns, or executed missions in a localized war.

1.6 General Support

This project was developed using available AFIT computer hardware and software resources. Access to AFIT's VAX 11/780 computer and Z-80 microcomputers was required. Previous experience, acquired background, academic support, current literature research, advisors' orientation, the provided BAF systems specifications, and the previously mentioned BAF file management system were the basic support for this research project.

1.7 Order of Presentation

Following this introductory Chapter, where the problem is identified and described, Chapter II explains all levels of system requirement specifications, what needs to be done for this thesis work, and why (14).

Chapter III, System Analysis and Database Design Approach, analyzes the system environment using the Decision Support System (DSS) Theory, describes a DSS architecture approach, and defines the database design approach and methodology to be developed. Chapter IV, Database System Design, follows the methodology stated in Chapter III and describes the database design process. It starts by constructing top-down plans, then develops a conceptual design and a logical model design.

The BAF Flying Unit Operational Control (BAF FLUNITOC) System is partially implemented in three different levels in Chapter V. On the first level, a prototype of a DSS Dialog Generator Management Software is built through the development of top-down menu driven programs using the dBASE II data base management system (DBMS), version 2.1, running on a Z-80 microcomputer. On the second level, a sample of a database system Main Menu and a Report Generator application program is implemented using the INGRES DBMS, version 7.10, and the "C" language, running on a UNIX VAX 11/780. On the third level of implementation, some advanced queries are retrieved from the designed relational database. In order to optimize future query retrievals from the implemented database, an investigation of optimum query retrievals is performed in Chapter VI. A hypothetical front-end system for the designed BAF database using natural language is roughly specified through some examples. Finally, trends and future directions for an optimum solution are discussed using Artificial Intelligence (AI) methods and techniques. Chapter VII summarizes the research findings, and states recommendations and conclusions (13).

II. Requirements Specifications

Before starting the requirements specification for the new sytem, it is necessary to briefly summarize the current BAF file management system.

As stated in Chapter I, the BAF already has a partially implemented file management system that manages personnel, material, operations, and specific information directly related to the air activities of Flying Units (Squadrons and Groups). Basically, the system supports tasks of controlling information of: crew members, aircraft, flight missions, and also specific information generated during the time to perform flight missions, e.g., time interval between aircraft engines start-up and cut-off.

The current file management system, shown in Figure 1, has many advantages when compared to the previous manual system, such as, improved response to flying unit management requests, improved report accuracy, and decreased bureaucracy. Although better than the manual system, the file management system has some technical and serious problems. For instance, the large amount of redundant stored data causes a waste of storage space because each Squadron or Group has to have its own files. Considered time consuming, file debugging application programs had to be specially designed, in order to avoid data redundancies and inconsistencies when users' requests increased (11).

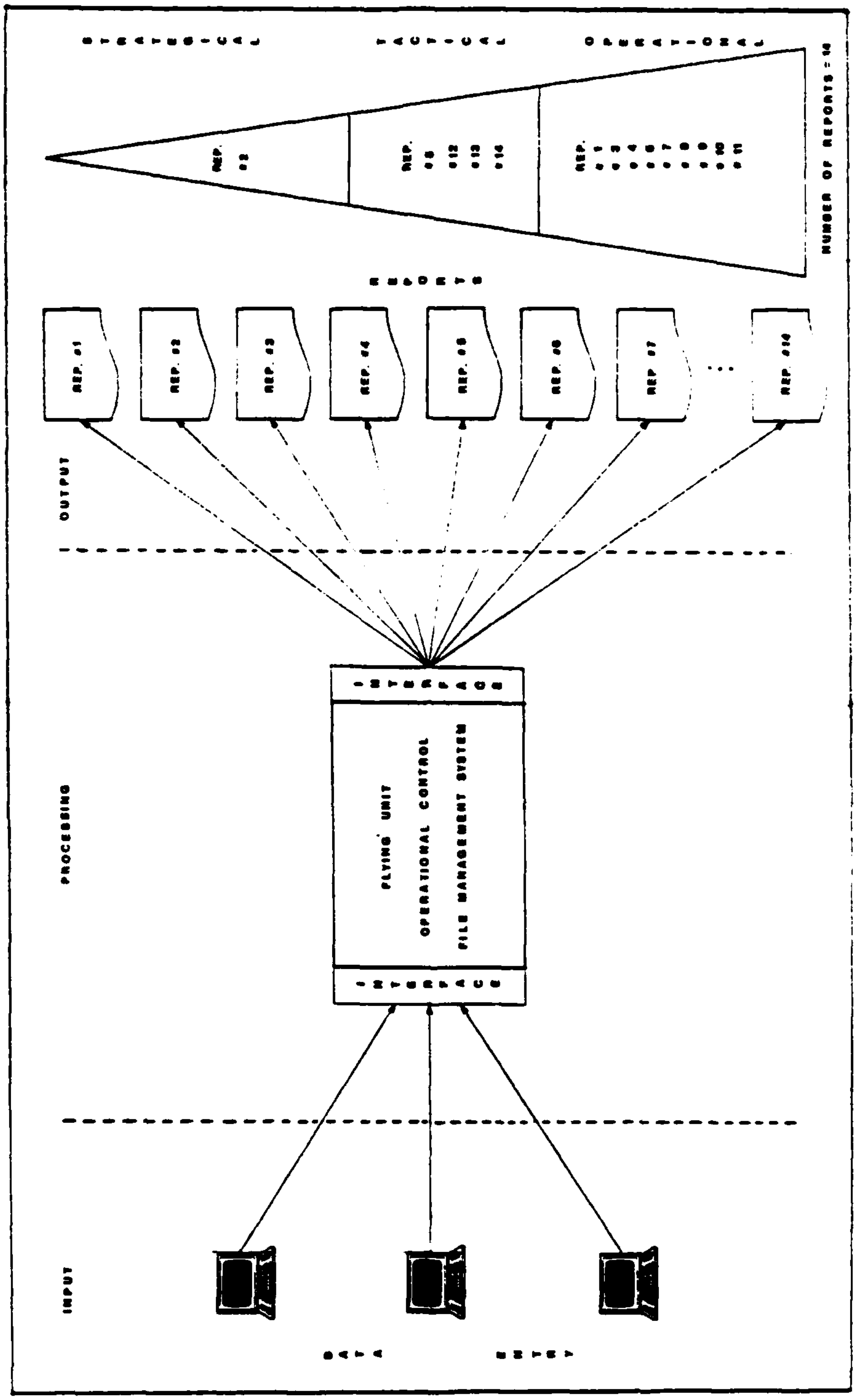


Figure 1 - The Current File Management System

Frequent updating procedures dealing with continous changes and many other additional application programs also had to be developed for the current file management system, generating more and more maintenance efforts. Consequently, a new data processing system avoiding the existing file management system problems and limitations is required for the BAF.

This Chapter defines requirements specifications in the following four levels: general system requirements, specific system requirements, database system requirements, and finally the thesis requirements.

2.1 General System Requirements

In order to better understand the main characteristics of the BAF system environment, before starting the specific system requirements, it was determined that a general planning of systems for the BAF should be designed, not only to help project new systems developments, but also to regulate the systems' growth (3) (25). Initially the following general system requirements were specified:

a) a new system should be designed to increase the efficiency of the current BAF current file management system;

b) the new system should be defined and designed in the context of a general top-down planning for systems;

c) the new sytem should be designed to store, process, and retrieve information about flying unit operations faster and more efficiently;

d) the new system should be an integrated data processing, reducing redundancies, avoiding data file inconsistencies and providing shared data among different users, in the sense that each user may have access to the same piece of data and may use it for different purposes at the same time;

e) the new system should provide centralized data control, enforce standards to be followed, use security restrictions, maintain data integrity, and avoid conflicts with the system requirements; and

f) the new system should be user-friendly, performing tasks with as much data independence as possible;

g) finally, without drastically impacting application performances, the new system should permit more flexibility and cheaper changes in application programs, in storage structures (physical records), and in access strategy.

2.2 Specific System Requirements

In addition to future user requests, it was determined that the new system should be able to interactively and concurrently perform the following specific requirements:

a) process data entries;

b) process more efficiently and in a flexible manner the current flying unit file management system reports listed below and described in Appendix A:

1) Individual Flight Record,

2) Mission Type Summary,

3) Crewmember's Summary per Aircraft Type,

4) Missions Summary per Aircraft Number,

5) Missions Summary per Administrative Unit,

6) Mission Orders Numbers List,

7) Consumed Items per Mission,

8) Aircraft Numbers Summary Totals,

9) Aircraft Numbers Status,

10) Consumed Items per Aircraft,

11) Consumed Items Quantity,

12) Crew Member's Summary,

13) Crew member Totals, and

14) Crew member's Totals Summary.

c) process update transactions in order to minimize maintenance costs; and

d) be able to deal with some formulated operational queries.

2.3 Database System Requirements

In order to satisfy general and specific system requirements, it was determined that the design of a database application system should be performed. Besides centralized data control, this database system should provide many advantages over the current file management system such as: minimize redundant data, avoid data inconsistencies, provide data sharing, maintain data integrity, control security restrictions and provide as much data independence as possible. The users of this new system should be able to provide data to be stored in the database. As decision makers, they should also be able to use information obtained by interactively accessing the database.

From the users' point of view, the following conditions should be met: today's needs for information should be satisfied in a reasonable time; anticipated and unanticipated end user's requirements should be satisfied; the database model should be easily expandable if necessary or modifiable in case of software and hardware changes; data integrity and data validity checks should be provided; and finally only authorized people should have access to data stored in the database (12).

2.4 Thesis Requirements

As a consequence of the previous requirements, it was determined that this thesis work should address the following issues:

a) an environmental system analysis able to produce a top-down system and database system plans for the BAF system environment;

b) a database design for a BAF Flying Unit Operations System;

c) a database partial implementation for a BAF Flying Unit Operations System; and

d) testing on some representative database queries.

It was assumed that this thesis research could be used later as a guide line for designing future database application systems for the BAF operations system environment. Consequently, the database design process should be as complete as possible, in order to create a general model independent of any database management system.

It was determined that the main concern of this research should be the database design. A complete system implementation is not required, because it is beyond the scope of a thesis effort and also it should be developed and tested in the BAF operational environment. Instead, the presentation of the database design development process with a partial implementation is required.

In designing the first database for a BAF Flying Unit, this thesis effort should represent an attempt to satisfy these requirements (8:1,3).

III. System Analysis and Database Design Framework

Designing a system able to efficiently process flying unit operations for BAF is the major goal of this thesis work. In order to accomplish this goal, before starting the actual database design, an environmental system analysis should be performed and a database design approach and methodology defined.

3.1 Environmental System Analysis

To better understand the environment under which a BAF flying unit operations system should be designed, this section presents a top-down macro-environment system analysis and the DSS theory. Some of the DSS concepts introduced are applied to derive a DSS architecture model, from which, a prototype Dialog Generator Management Software (26) is projected and later implemented in Chapter V.

3.1.1 The Macro-environment Analysis

In order to produce a database top-down plan for the BAF system environment, as specified in Chapter II, a macro-environment system analysis should be performed first. To start analyzing the macro-environment, the BAF was initially considered as a system, that is, a group of individual components working together to form a unified whole. Assuming that any system can be a component or a sub-system of another system, the BAF system was considered as a component or a subsystem of a major system identified as the National Defense System. The BAF was also considered at the same level of the Brazilian Army (BARMY), Brazilian Navy (BNAVY), or any other national civilian department, as for example, Brazilian Economics System.

As shown in Figure 2, the BAF System was also decomposed into several sub-systems, as for example, the BAF Operations System, BAF Logistics System, BAF Training System, etc.. It was noticed that these systems could be considered as parts of a general corporate system or as different corporate sub-systems, because the BAF system and its subsystems essentially differ from each other by their degrees of corporate data aggregation handled. The higher the system level, the higher became the degree of its data aggregation.

Using the same previous criteria, a BAF Operations System, considered the main interest of this thesis work, was initially identified as a BAF subsystem and decomposed in several sub-systems such as, the Flying Unit Operations System, Administrative Unit Operations System, Training Unit Operations System, etc. To better understand and locate a Flying Unit Operations System in this environmental structure, a pictorial representation of the macro-environment is shown in Figure 2, also supported by the DSS Theory explained in the next section.

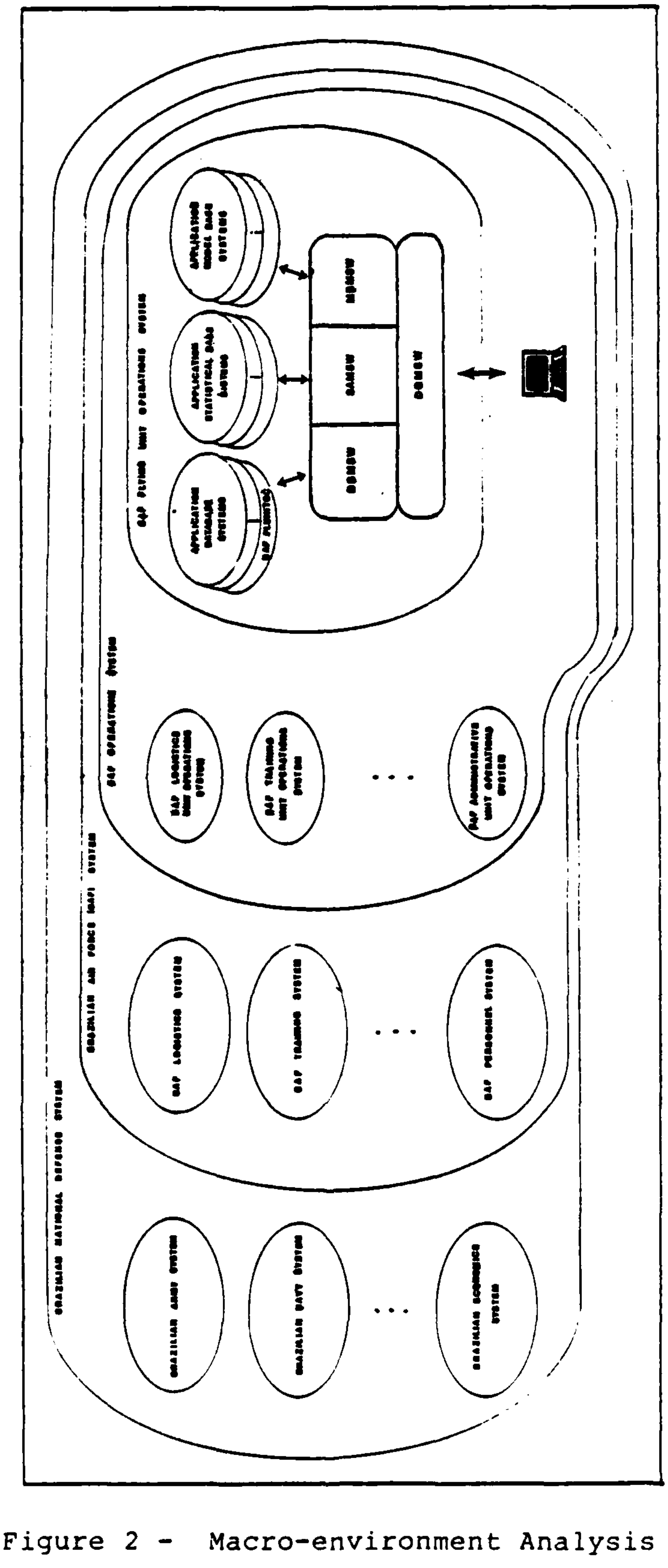


Figure 2 - Macro-environment Analysis

3.1.2 Decision Support System (DSS) Theory

In order to support decision making processes at all the different levels, it was decided that for each identified system, a correspondent Decision Support System should be dimensioned. A Decision Support System (DSS) refers to an interactive computer-based system designed to help decision makers decide about data to solve unstructured problems (26).

It was noticed that, when the systems were broken down, their levels of complexity reasonably decreased. When a system was characterized by some specific goals, functions or tasks that were easy to understand, then it was identified as a specific system, that is, a system simple enough to be handled by a human being that could be considered as a self-contained application system. An application system refers to a related group of application programs. For instance, a Flying Unit Training System could be composed of a dozen different application programs that are run at different times for various purposes. Collectively, all these programs could contribute to the same decision makers, dealing with the training function of a specific Flying Unit Training System. An application program refers to a computer program for a given user that solves a specific problem or performs specific actions.

The DSS Theory focuses on top managers and executive decision makers, emphasizing the needs for flexibility, adaptability, quick response, and the ability to support personal decision-making styles of individual managers. The theoretical framework of a DSS is helpful in analyzing a complex system environment, identifying the relationships among the system components, and revealing system areas in which further developments should be required (5).

Basically, three levels of hardware/software have been included as parts of the Decision Support System Theory. They are used by people with different levels of technical capability. They also vary in nature and scope of task to which they can be applied. The relationships between these three levels are illustrated in Figure 3.

Specific Decision Support Systems are the first level of hardware/software that allow a specific decision maker or group of them to deal with specific sets of related problems, as in the previous mentioned example of a Specific Flying Unit Training System. DSS Generators are considered in the second level as a package or group of related hardware and software, which provides a set of capabilities to build a specific DSS. For instance, a Specific Flying Unit Operations System could be built through menu driven programs with a set of integrated capabilities, including data-entry, report generation, inquiry transaction, modeling language, graphic display, and statistical analysis (26).

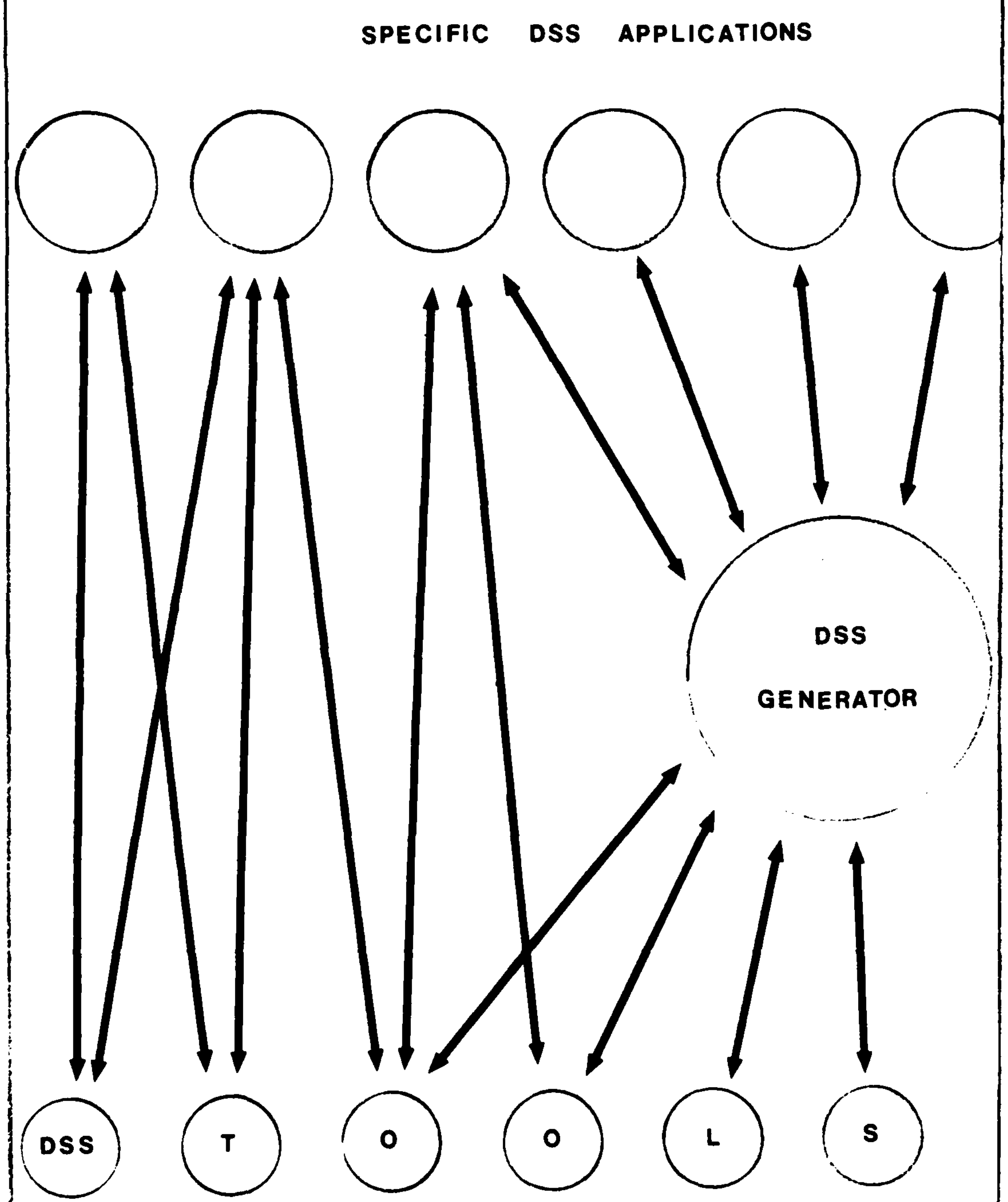


Figure 3 - Three Levels of DSS (26).

Such a system, once generated, could be replicated with minor changes in the whole package. In fact, from this replication idea came the name DSS Generator, considered in the second level of DSS. The result is that a Specific Flying Unit Operations System could be used as a DSS Generator. In other words it could be replicated specially for a Specific DSS with the same characteristics, having decision-making operations functions, as for example, a Specific Administrative Unit Operations System or a Specific Logistics Unit Operations System (6).

Decision Support System Tools are considered in the third and most fundamental level of technology applied to the development of Decision Support Systems. In this level, there are hardware and software elements which facilitate the development of Specific DSS or DSS Generators. This category of technology has seen the greatest amount of recent development, including new special purpose languages, improvements in operating systems to support conventional approaches, data base management systems (DBMS) and supporting software. All these three levels may be better visualized through an example following a DSS Architecture Model Approach explained in the next section.

3.1.3 The DSS Architecture Model

Employing the DSS theory introduced in the previous Section, a DSS Architecture Model shown in Figure 4 was adopted. As an example to better understand the DSS theory application, it was assumed that a Specific Application System from a Flying Unit Operations environment could be developed as a Specific DSS, using the concepts of a DSS Generator and DSS Tools. In this case, specific features of data base management software, statistical base management software, and model base management software, could be considered as DSS Tools (26).

Some available features to be used as a data base management system (DBMS) are TOTAL, INGRESS, dBASE II, and RBASE, which could be used as data base management software, integrated through a DSS Generator called Dialog Generator Management Software (DGMSW). In this thesis, only INGRESS and dBASE II DBMS are used. Some available features of statistical packages, as for example, "S" and "SPSS", could be used as statistical base mangement software and integrated through the DGMSW DSS Generator. Finally, some available features of modeling packages as for example, "SLAM - Simulation Language for Modeling" and "GASP IV - Simulation Language", could be used as a model base management software and also integrated through the DGMSW DSS Generator. All these issues are illustrated in the DSS Architectural Model shown in Figure 4. The appropriate understanding of the DSS Architecture Model Approach using the DSS theory was an important prerequisite to the environmental system analysis.

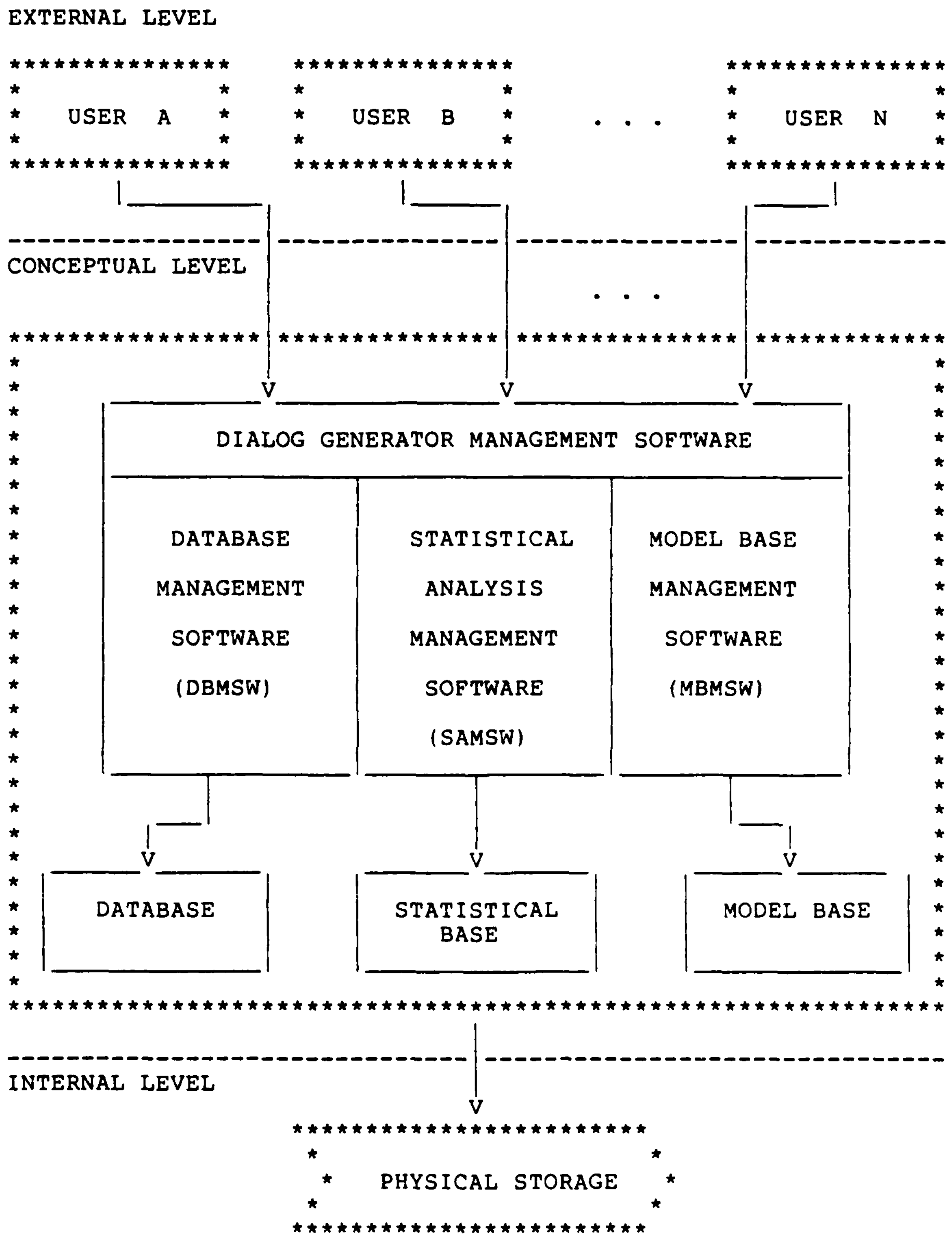


Figure 4 - The DSS Architecture Model

The feasibility of this architecture model has been demonstrated through a Management System Analysis and Simulation course project (10). Only parts of the DSS Architecture Model are addressed in this thesis. Following this architecture model, a prototype Dialog Generator Management Software is developed and implemented in Chapter V. Using INGRES DBMS, a database is designed in Chapter IV and partially implemented in Chapter V as a Specific DSS under a Flying Unit Operations System.

Following the previous example, a Specific Flying Unit Operations System could be developed as a DSS Generator. Specific features of database management software, statistical base management software, and model base management software could be also used as DSS tools (26).

3.2 Database Design Framework

In this section the database design theory is introduced and the design approach and methodology are developed, in order to support the actual database design performed in Chapter IV.

3.2.1 Database Design Architecture

To design a database an architecture model should be chosen. For this database design the model chosen is shown in Figure 5 divided into three general level: external, conceptual, and internal (12).

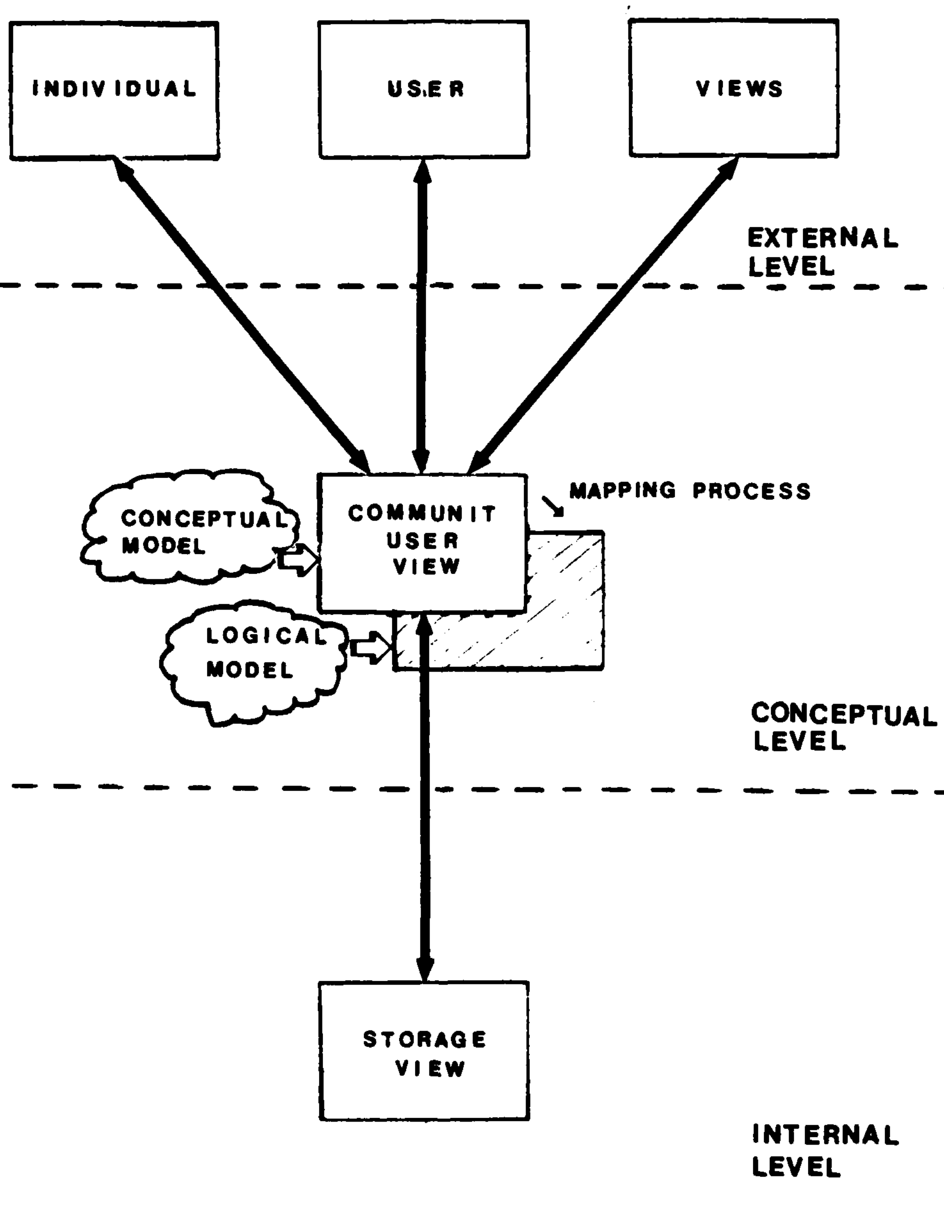


Figure 5 - The Database Architecture Model (12).

The external level is characterized by the external or individual user's views of the data, the one closest to the users and concerned with the way in which data is viewed by individual users.

The conceptual level may be thought of as a community user view of the data, in other words, concerned with the ways in which data is viewed by the Database Administrator (DBA). The DBA refers to an individual or a group of individuals with an overview of one or more database applications, who controls the design and use of these databases. It is often better to use two individuals or groups, a DBA and a Database Designer who designs the physical aspects of a database.

The internal level is the one closest to physical storage and concerned with the way in which data is actually stored.

3.2.2 Database and Relational Theory

In order to provide enough background for a database design process, this section introduces the necessary theoretical concepts. The relational concepts presented here support the following sections of this chapter.

The term database refers to a repository for any integrated and shared stored data. The term integrated means the unification of several otherwise distinct data files, with any redundancy among files partially or wholly eliminated. The term shared means that individual pieces of data in the database may be shared among several different users, in the sense that each user may have access to the same piece of data and may use it for different purposes (1).

A data base management system (DBMS) facilitates use of the database. A DBMS refers to the software required for using a database, and presenting multiple different views of the data to users and programmers. A DBMS uses a data model as its underlying structure. There are three different types of data models: hierarchical, network, and relational. The main difference among the three lies in the representation of the relationships between entities (23). A brief comparison between data models is presented in Section 3.2.3.

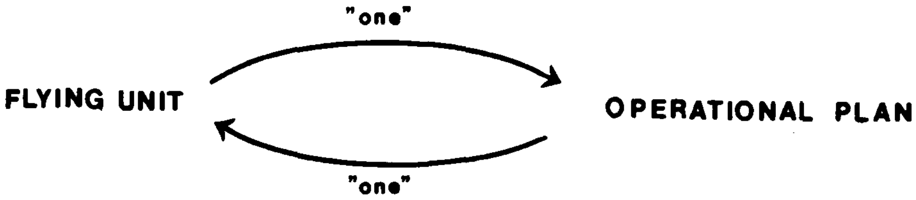
For this thesis, the concepts associated with relational data models and relational databases became particularly important. The major concept from the relational data model used in developing the conceptual model of a database design is the normalization process, that is, the process of grouping data items into tables representing entities and their relationships (7). Using relational database terminology, a table with rows and columns is named a relation with tuples and attributes. An entity is considered a person, place, thing, or concept that has characteristics of interest to the organization, in other words, it is something about which data is stored. A data item refers to the smallest unit of data that has meaning in describing information. It has the same meaning as data element or field (1).

Considering its importance for understanding a data model and especially a relational data model, the following concepts need to be explained: a) relatioships within a data model, b) the relational data model, c) tabular repesentation, d) keys, e) data dependency, and f) normalization process.

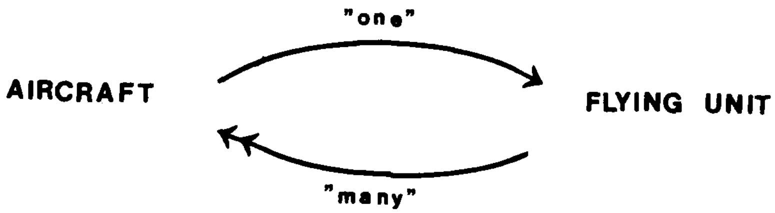
a) Relationships within a Data Model

Relationships may exist within a data model and because of its importance for a relational data model, concepts involving relationships are explained as follows. A relatioship is a mapping or linkage between two sets of data that can be divided in three types: one-to-one, one-to-many, and many-to-many. Relationships within a data model may exist between entities, between attributes of the same entity, or between attributes of different entities. For example, to accomplish missions projected in the BAF operational planning, a flying unit is composed of several aircraft and crew members able to perform those misssions. In this case, some of the entities can be: missions, BAF operational planning, flying unit, aircraft, and crew members (1).

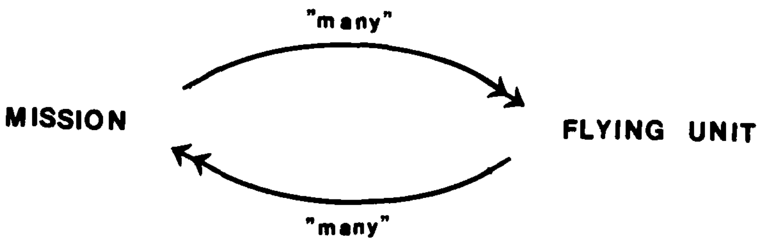
Considering relationships between entities, if at a given point in time, one FLYING UNIT may have only one OPERATIONAL PLAN and vice-versa, this characterizes "one-to-one" relationships. "One-to-one" relationships can be denoted by single-headed arrows, as shown in Figure 6(1). If at a given point in time, zero, one or many AIRCRAFT are assigned to one FLYING UNIT, but an AIRCRAFT is assigned to only one FLYING UNIT, this characterizes "one-to-many" relationships.



(1) "One-to-one" entity relationships.



(2) "One-to-many" entity relationships.



(3) "Many-to-many" entity relationships.

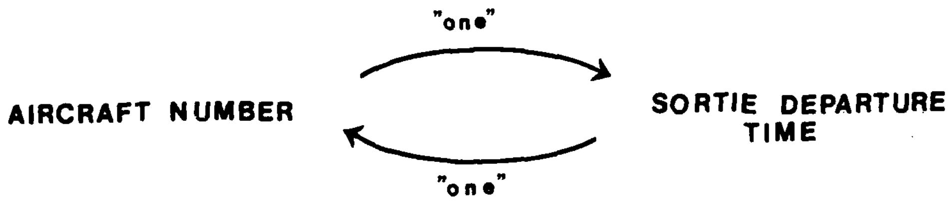
Figure 6 - Relationships between Entities (1).

"One-to-many" relationships can be denoted by a single-headed arrow going in the "one" direction and a double-headed arrow going in the "many" direction, as shown in Figure 6(2). In the example, a FLYING UNIT may have operated on several MISSIONS, or a MISSION may have been conducted by several FLYING UNITS. In this case, the relationship between FLYING UNIT and MISSION is characterized as "many-to-many", as shown in Figure 6(3).

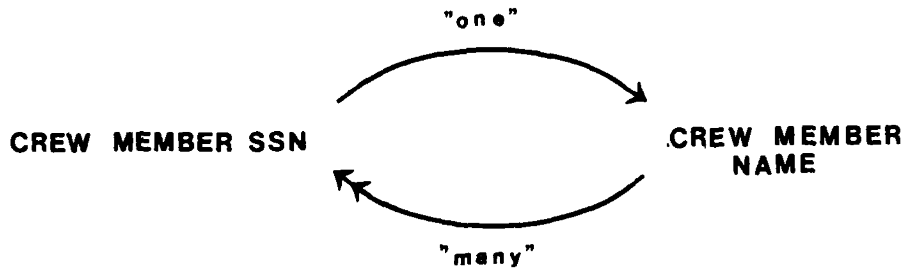
The relationships between entities are part of the conceptual model, and they have to be represented in the database. The same entities can participate in any number of relationships, and on the other hand, any number of entities can participate in a relationship (1).

Relationships between two attributes of an entity are also classified as "one-to-one", "one-to-many", or "many-to-many". The AIRCRAFT TAIL NUMBER is a unique identifier of an aircraft, that is, the AIRCRAFT NUMBER is an attribute that uniquely identifies an aircraft entity. If together with the AIRCRAFT NUMBER, another unique identifier of the aircraft is stored in the database, the relationship between the two identifiers is "one-to-one", and it also can be denoted by single-headed arrows, as shown in Figure 7(1).

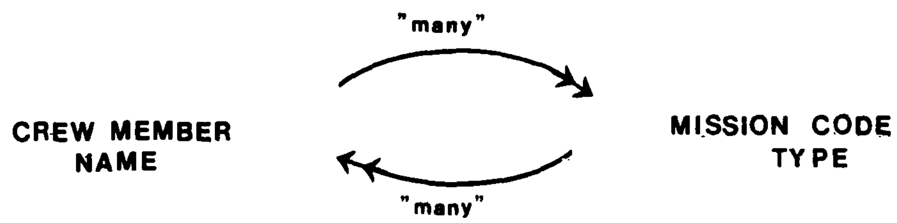
Considering the crew member entity, if the LAST NAME and SOCIAL SECURITY NUMBER (SSN) exist as its two attributes, there can be many crew members with the same name, but with different SSN.



(1) "One-to-one" attribute relationships.



(2) "One-to-many" attribute relationships.



(3) "Many-to-many" attribute relationships.

Figure 7 - Relationships between Attributes (1).

Every crew member has a unique SSN, that is , to a given crew member SSN there corresponds only one crew member name. Here, once more, the "one-to-many" relationship can be denoted by a single-headed arrow in the "one" direction (crew member SSN) and a double-headed arrow in the "many" direction (crew name), as shown in Figure 7(2).

If a number of CREW MEMBERS with the same NAME may perform many MISSIONS, and a number of missions with the same code-type-name may be flown by many crew members, the relationship between the attributes crew member's name and missions' name is "many-to-many". This relationship is denoted by double-headed arrows, as shown in Figure 7(3).

b) <u>The Relational Data Model Approach</u>

Since the early 1960s the hierarchical and network data models have been in use as structures for database management systems. The relational data model has been used since the early 1970s. Basically, the main difference among the three data models is the way they represent the relationships of entities. Hierarchical and network data models use pointers or links to handle and represent data relations, causing restrictions to many data changes needed during the database growing. As soon as a new user's request arrives, the number of logical pointers increases together with the number of application programs needed, causing changes in the logical database representation (4). Consequently, the level of complexity of these models increases more and more.

The relational data model can be easily understood by a user without training in programming, can avoid changes in the logical existent data model structure while the database is growing, and permit a desirable amount of flexibility in formulating unanticipated queries. The following concepts under a relational data model represent the basis of the general design methodology used in this thesis work (1).

c) Tabular Representation

In the relational data model, entities and their relationships are represented with two-dimensional tables. A two-dimentional table is called a relational model of the data. In a relational data model terminology, a table is called a relation. To avoid confusion between a relation and a relationship between the entities, a relation is sometimes called a table. Every column in a relation is an attribute. The values in the column are drawn from a domain. A domain refers to a set of all the values an atribute may have. For example the domain for aircraft tail numbers is made up of all four-digit integers from 0000 to 9999 but the actual aircraft tail numbers may be only 2120, 2123, and 2122. The rows of the table are called tuples. Using a conventional terminology, the columns of a table can be considered as data items, and the rows, as data records. In a relational data model, relationships are also considered as entities, and every table represents an entity (12).

Tables are also characterized by having each entry representing a data item, and both rows and columns can be viewed in any sequence at any time without affecting the semantics of any function using the table or the information contents. A description of keys and data dependency will be presented, in order to motivate the concepts of a normalization process, the most important support to the relational data model approach.

d) Keys

Frequently, within a given relation there are one or more attributes with values that are unique within the relation and thus can be used to identify the tuple of the relation. This value may be used to distinguish that tuple from all others in the relation, and is called primary key. If a primary key consists of only one attribute, it is called a simple primary key, and if it consists of more than one attribute, it is called a composite primary key. The other data items not considered as primary key are called nonprime attributes. It is assumed that the primary key is nonredundant, in the sense that none of its constituent attributes is superfluous for the purpose of unique identification. If there is more than one attribute combination needed to process this unique identification, this combination is called a candidate key. One particular candidate key is selected to be used as a primary key, and the others are called alternate keys.

e) Data Dependencies

Different types of dependencies may exists between the data items. For the normalization process, each type of dependence, explained as follows, has to be taken into consideration. The fundamental concept of functional dependence (FD) refers to a set of attributes "B" of a relation "R", which is functionally dependent on another set of attributes "A" of a relation "R". If at any instant of time, each value of "A" has one and only one value of "B" associated with it, then given any value of "A", the value of "B" is uniquely determined, and the convention A ---> B is adopted, meaning that "A" uniquely determines "B" and "B" is <u>functionaly dependent</u> on "A".

Consider the same two sets of attributes "A" and "B" of a relation "R", if the set "B" is functionally dependent on "A" but not functionally dependent on any proper subset of "A", the set of attributes "B" is said to be <u>fully functionally dependent</u> on set "A".

Consider three sets of attributes "A", "B", and "C" of the same relation "R", if A ---> B and B ---> C, then implicitly A ---> C and consequently at the same time , B -/-> A, meaning that "B" can not determine "A". In this case a set of attributes "C" of the relation "R" is said to be <u>transitively dependent</u> on "A" (1).

f) <u>Normalization Process</u>

If a relational data model design approach is adopted, the inputs for the design process are the data items and the data dependencies previously discussed. Grouping these data items into a set of relations, constitutes the framework of a relational database design. The normalization process is the discipline of grouping data into a collection of relations used during the design of a relational data model. The normalization theory is based on the observation that a certain set of relations have better properties in an inserting, updating, and deleting environment than other sets of relations containing the same data.

The reason for using the normalization procedure is to ensure that the conceptual model of the database will work without causing problems when application programmers attempt to modify the database. An unnormalized data model consists of records as they are used by application programs. In order to perform the normalization process, database designers should start with the user's views, usually roughly unnormalized data item groups, and then perform the normalization process step-by-step. Each step is called a specific normal form and produces a set of relations that has better properties than the previous one.

The first step in the normalization process is called first normal form (1NF) and consists of transforming the data items into a two-dimentional table, removing all of repeated occurrences of data items so that a flat file is obtained containing only atomic data values.

The second step in the normalization process is called second normal form (2NF) and consists of determining what the keys are and how the data items relate to the keys. While in the 1NF a tuple or entire row of the table is independent of all the key items, in the 2NF an attempt is made to determine what data items are related to parts of the total key. If data items depend only on part of the key, the key and the items connected to the partial key are candidates for removal into separate records. The 2NF refers to the process of breaking apart the first normal table into series of tables, in which each item depends only on the entire key.

The third step in the normalization process is called third normal form (3NF) and consists of separate data items from the second normal relations that, while dependent only on the key, may have an independent existence in the database. Therefore information about the data items can be entered separately from the relationships in which they are involved. A relation is said to be in the 3NF if and only if it is in the 2NF and every nonkey attribute is nontransitively dependent on the primary key.

Providing sucessive improvements in the insertion, deletion, and update operations against the database, the 1NF, 2NF, and 3NF constitutes the core part of a relational data model design, characterizing the entire normalization process, making the designers understand the semantics of attributes and their relationships, and ordering the thought process for data analysis (1).

3.2.3 Database Design Approach

After the initial phases of system definitions, requirements specifications, and general system analysis, there are two basic approaches to design database applications, as shown in Figure 8.

Considering the first approach, a Specific Database Model Design should be taken to be directly designed, using one of the three data models: relational, hierarchical, or network. In the relational data model, entities and relationships between entities are represented by tables or relations composed by rows or tuples and columns or attributes. In the hierarchical and network data models certain relationships are represented by means of links. Such links are indexes capable of representing mainly one-to-one and one-to-many associations. The basic difference between the hierarchical and network data models is that in the network, links may be combined to model more complex many-to-many associations, whereas this is not possible with the hierarchical data modeling (1).

In the second approach to design database applications, also shown in Figure 8, after the initial phases, a general or Canonical Database Model Design phase should be performed. Considering this approach, after each entity and relationship is determined, the conceptual model can be mapped to one of the three existing data modelings: hierarchical, network, or relational. A general or canonical form is desirable mainly because it is DBMS independent (1).

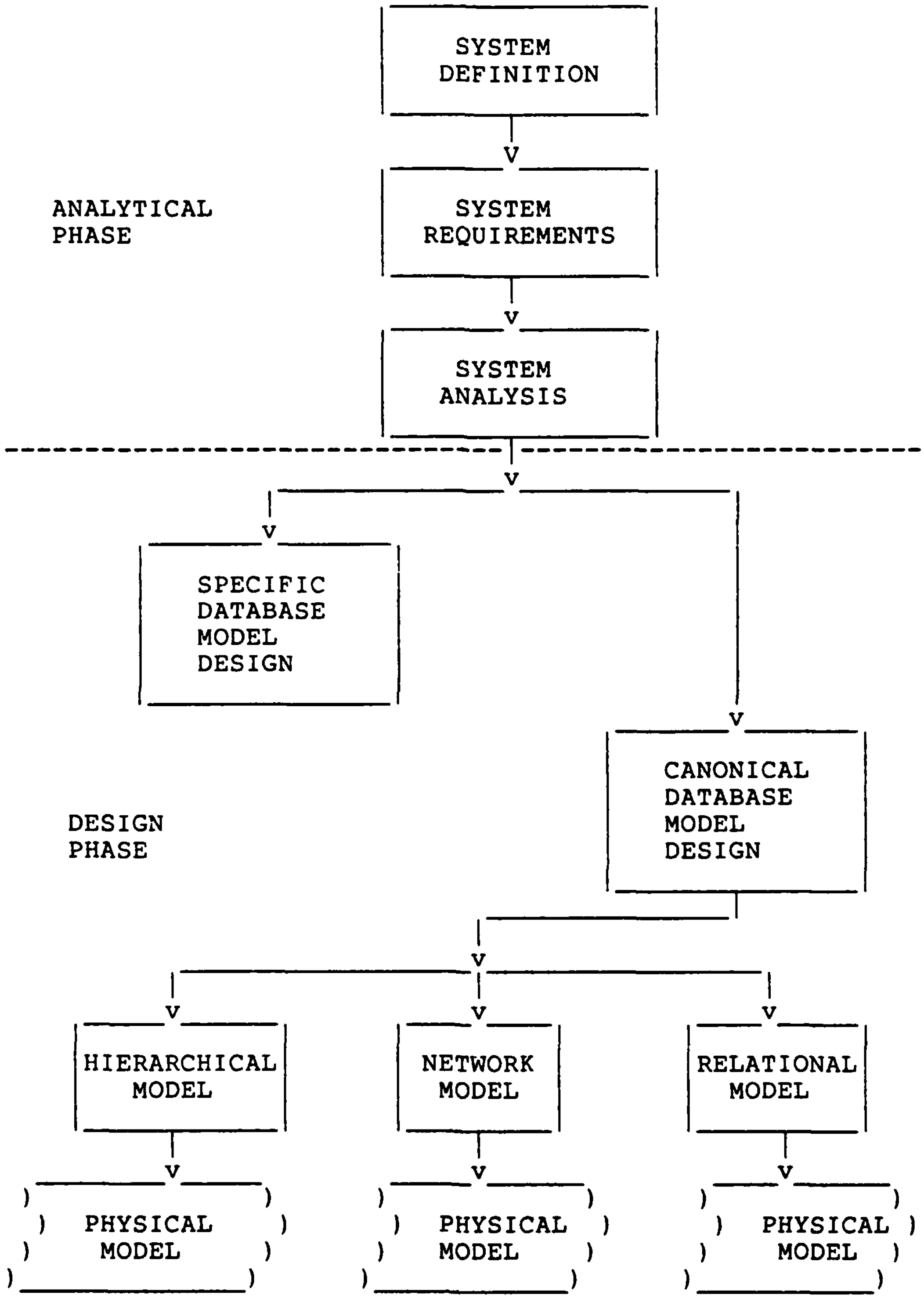


Figure 8 - The Database Design Approaches

A canonical approach can also be easily mapped to any of the three existing data modellings almost immediatelly after a DBMS has been chosen.

Because it is a more rational, well-defined, independent and structured design technique, the second approach, Canonical Database Model Design, was chosen to be adopted for this database design.

3.2.4 Database Design Methodology

To design a database application using the database architecture model stated in the previous section, four basic steps need to be performed: a) top-down system plans, b) conceptual model design, c) logical model design, and d) physical model design.

a) Top-down System Plans

In the top-down system plans, each part of the system, in this case a Flying Unit Operations System, should be analyzed and a model composed of functions and process should be developed. From this model, entities about which data are stored should be identified. Then an entity relationship chart should be derived and potential database applications determined by grouping entities that can be implemented in separate database applications. In order to increase the implementation speed of the first database, rough top-down system plans may be prepared at the beginning and refined after the development of each database application. Normally, the group of entities selected to be designed first should be one that is quick and easy to implement, solve immediate problems, and has fast payback.

b) Conceptual Model Design

The conceptual model design consists of building a model considering the entire number of user's views combined as a unique community user view, representing the conceptual contents of the database with all entities, relationships between them and data items. Despite data modeling independent, the conceptual model should be designed as a relational, hierarchical, or network model.

For the flying unit operations database design, the conceptual model should be designed using the relational data modeling concepts, as shown in Chapter IV. The normalization process previously described should be performed starting by determining the external or individual user's views of the data. First, all data items from the individual user's views should be described and integrated into a data dictionary. Then, all system assumptions required to generate the user's views should be stated. After that, all the relationships between the data items should be determined by identifying the key and nonkey data items. Then, for each user view, 3NF relations should be developed and where this is not possible for individual views, data from different user's views should be merged to stablish 3NF relations. Frequently, some relations are derived from more than one view, but each one has to be represented only once in the conceptual model.

In order to prepare an explicity graphical representation of the conceptual model, the 3NF relations derived from all user's views should be summarized in just one list separated by levels or number of primary keys. Representing entities, relations with only one primary key data item should be placed on the first level. Two primary key data item relations should be placed on the second level. In this case, if a part of a primary key is not represented as an entity, a new supporting entity relation should be generated on the first level. This procedure is repeated for each level that matches with the number of primary key data item relations. At the end of this process, if there were some relationships between the relations stated as assumptions in the beginning of the process, it should be included in the model. Finally, the generation process of the conceptual model based upon a chosen data modelling is considered complete. Afterwards, the conceptual model designed should be mapped to a relational, hierarchical or network data model (1).

c) Logical Model Design

After a conceptual model has been sucessfully designed, it should be mapped to a logical model, using one of the three data models: relational, hierarchical, or network. In order to select which one is the more appropriate data model to be used, some factors have to be taken in consideration. First, it was assumed that simplicity and flexibility from the user's point of view should be a relevant factor. This is especially important in a user environment with minimal database implementation experience. Second, it was determined that the data processing characteristics of the systems to be installed, should require an easy and more human-thinking-like data model structure. Finally, the data representation should be efficient. Considering that certain relationships can be better represented in one model than in others and comparing the three models to each other, the hierarchical model was discarded, because it is unable to handle many-to-many relationships, a constant and frequent characteristic of the systems to be installed. The network model was considered the most efficient, but its installation too complicated for the operational environment. The relational model was chosen because of its simplicity, flexibility and human-thinking relational type of structure (1).

Basically, because the conceptual model was developed using the relational approach, a relational logical model is easier and simpler to implement.

d) Physical Model Design

The last step of a database design, the physical model design, is concerned with the way data is stored on physical devices. The design of the physical model should be performed only after the DBMS has been chosen. Essentially, the physical aspects of the DBMS and the direct access device characteristics should be taken into consideration.

The physical model design is not addressed in this thesis work, because important information about the database size, volume and frequency of access are not available.

IV. The Database System Design

A good system design avoids excessive complexity. However, building a unique database system for a large organization is a complex task. Usually, a certain number of integrated databases should be designed, since it is far beyond the capability of any team to design just one database for the whole corporation. Even if it could be designed, machine performance considerations would make it unworkable (1).

In this chapter an application database system is designed for the BAF Flying Unit Operations, according to the problem defined in Chapter I, systems requirements Specified in Chapter II, and the DSS and database theory presented in Chapter III.

4.1 DSS and Database Systems Top-down Plans

After analyzing the entire system environment in Chapter III, system levels where decision making processes take place were better understood using the Decision Support System Theory. Before performing a top-down planning for databases, it was determined that, another top-down planning would be necessary to be perfomed for decision support systems. A graphical representation of decision support systems top-down planning is presented in Figure 9, where the Flying Unit Operations System was identified at the fourth system level, as a specific decision support system.

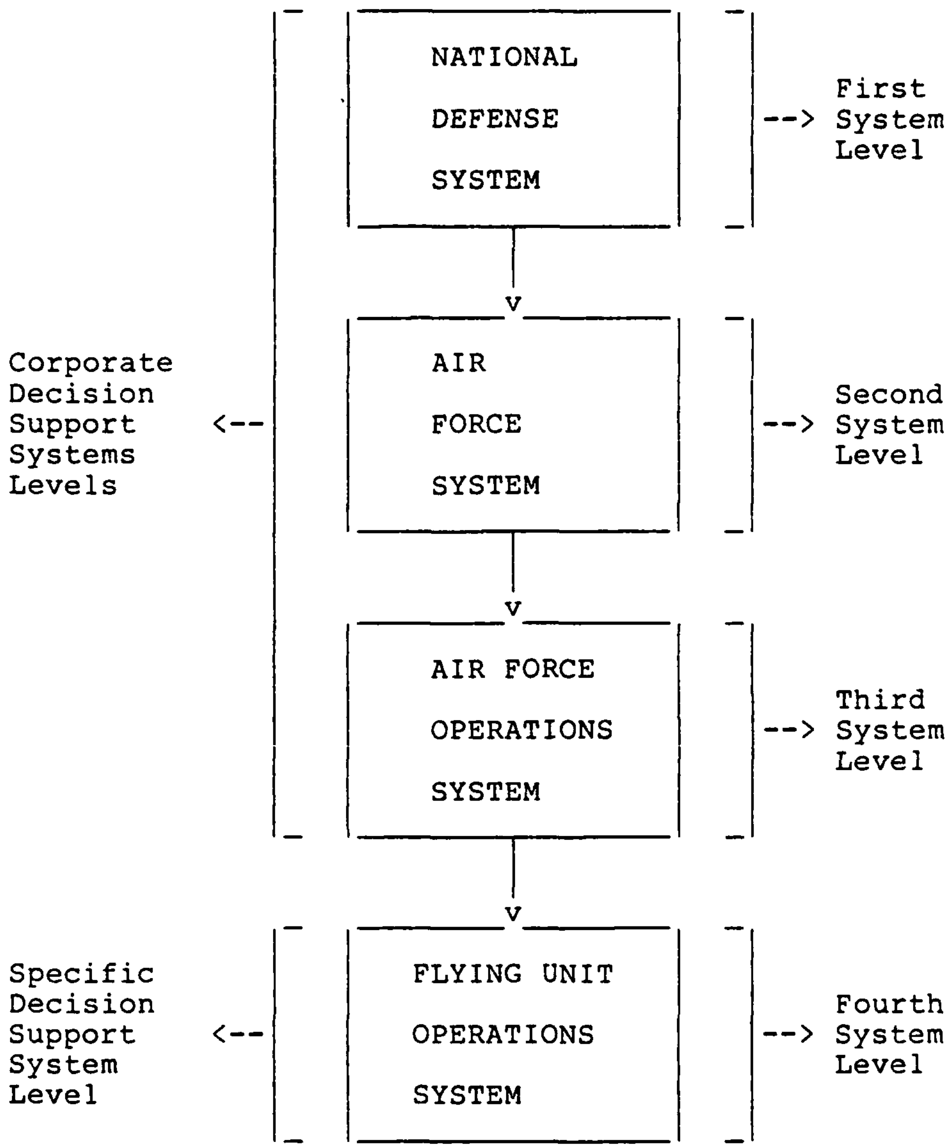


Figure 9 - The DSS Top-down Planning

Starting from the top, the National Defense System can be broken down on a second system level, where the Air Force System is located. An Air Force System also can be broken down on a third system level, where the Air Force Operations Control System is located. Finally, an Air Force Operations

Control System can be broken down on a fourth system level, where the Flying Unit Operations System is located (10).

Performing a top-down planning for database systems afforded a better understanding of the specific environment of a Flying Unit Operations System, as shown in Figure 10.

Every database system located above the fourth system level was considered as a part of a corporate database system. The Flying Unit Operations System by itself was considered as a specific subject database system, large enough to be broken down into a fifth system level, filling all the ideal conditions to be divided and conquered. It was also determined, that this fifth system level, from the database design point of view, could be called an application database systems level. At this level, specific functions or applications can be identified and treated singly. Having their own independent life cycles these functions or applications become easy to implement considering the amount of data items to be handled.

While performing the top-down planning for databases, it was desirable to identify those database systems for which a payback could be quickly demonstrated. The most important part of the top-down planning was the selection of the database systems implementation priorities. It was decided that the ones to be implemented first should be those which could solve immediate problems, have fast payoff, and could be quick and not so complex to implement.

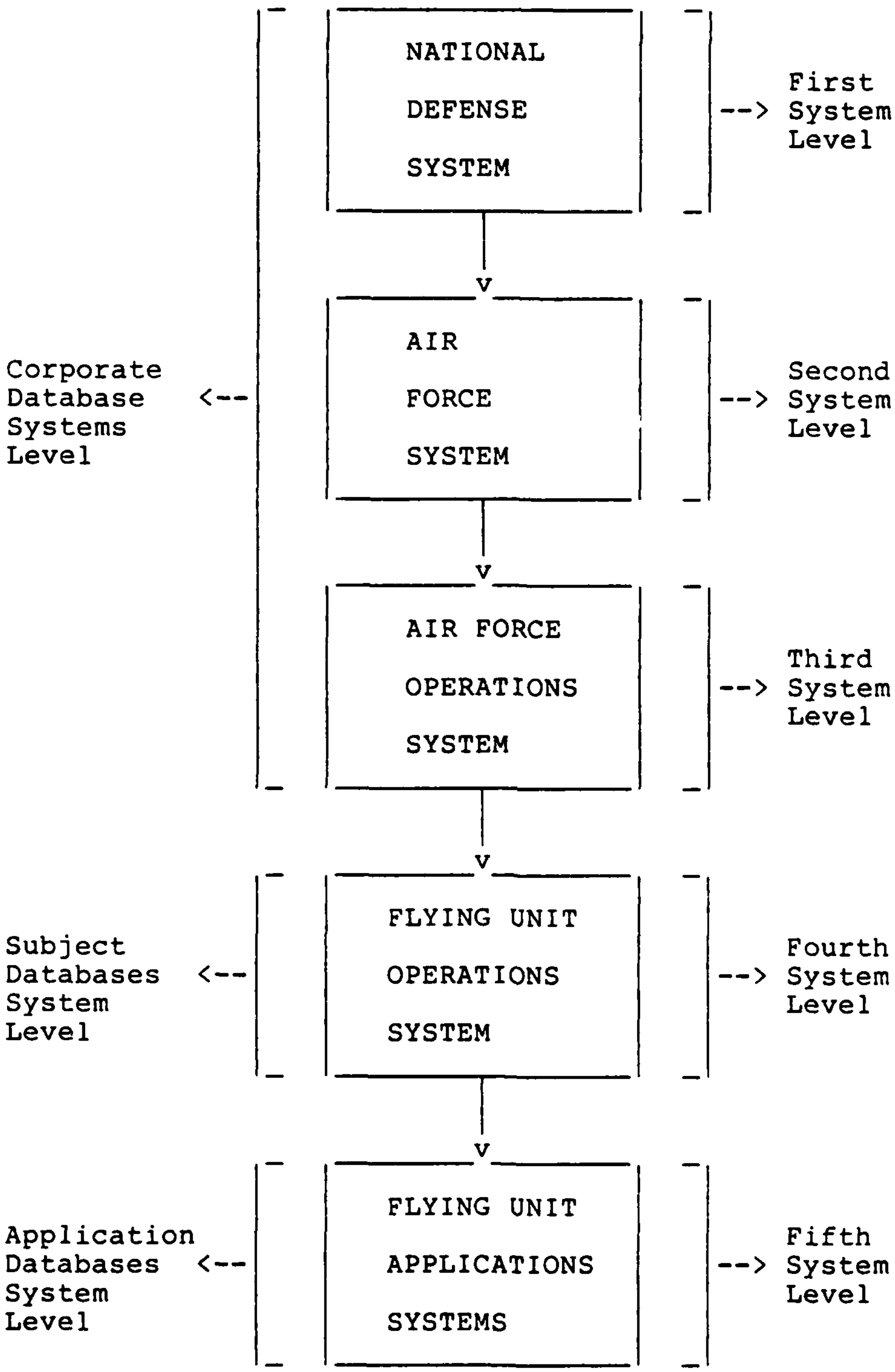


Figure 10 - The Database Systems Top-down Planning

Out of the flying unit operations environment, the Flying Unit Operational Control System was identified as one potential database application able to fulfill all these requirements.

The database systems top-down planning considered the National Defense System, the Brazilian Air Force System, and the Brazilian Air Force Operations System, as parts of a corporate database level. The Flying Unit Operations System was considered as a subject database level, where several application databases could be designed.

Implementable application database systems, distinctly developed as pieces of a Flying Unit Operations system subject database, could not only solve immediate flying unit problems, but could also tremendously increase the overall Air Force corporate efficiency by increasing the efficiency of each of its flying units.

When an overall corporate database level is considered too broad to form a database itself, subsets of it can be partitioned and become implementable databases. Generally speaking, an overall corporate database level can represent a model much broader in scope than any specific database and it can not be converted directly into a database conceptual model or schema. A database conceptual model is defined as a process of designing a model representing inherent properties of data of a subject database, independently of any software or hardware. Only extracts from a subject database, called application databases, become database schemas (21).

Multiple database conceptual models or schemas can be derived from a subject database level, and in turn, multiple subject databases can be derived from an overall corporate database level. The broader operation of building an entire subject database can be carried out later by synthesizing all useful application databases containing data items about a specific subject and synthesizing executed functions.

Application database levels can be considered, when data required for a specific application or group of applications are able to be synthesized from available documents or users' views. In all three different levels of database, data needs to run, an environment needs to be defined in a data dictionary, and data needs to be modeled a step at a time.

As shown in Figure 11, a database application level can be mapped to a conceptual model and afterwards to a logical model (hierarchical, network, or relational) (1).

An application database is considered the lowest level for a database design. Several application databases about a specific subject, function or field, can be integrated into a subject database level, and in turn, several subject databases can be integrated into a corporate database level. Following this aggregation criteria, future database needs can be projected and expanded, as soon as new needs arrive.

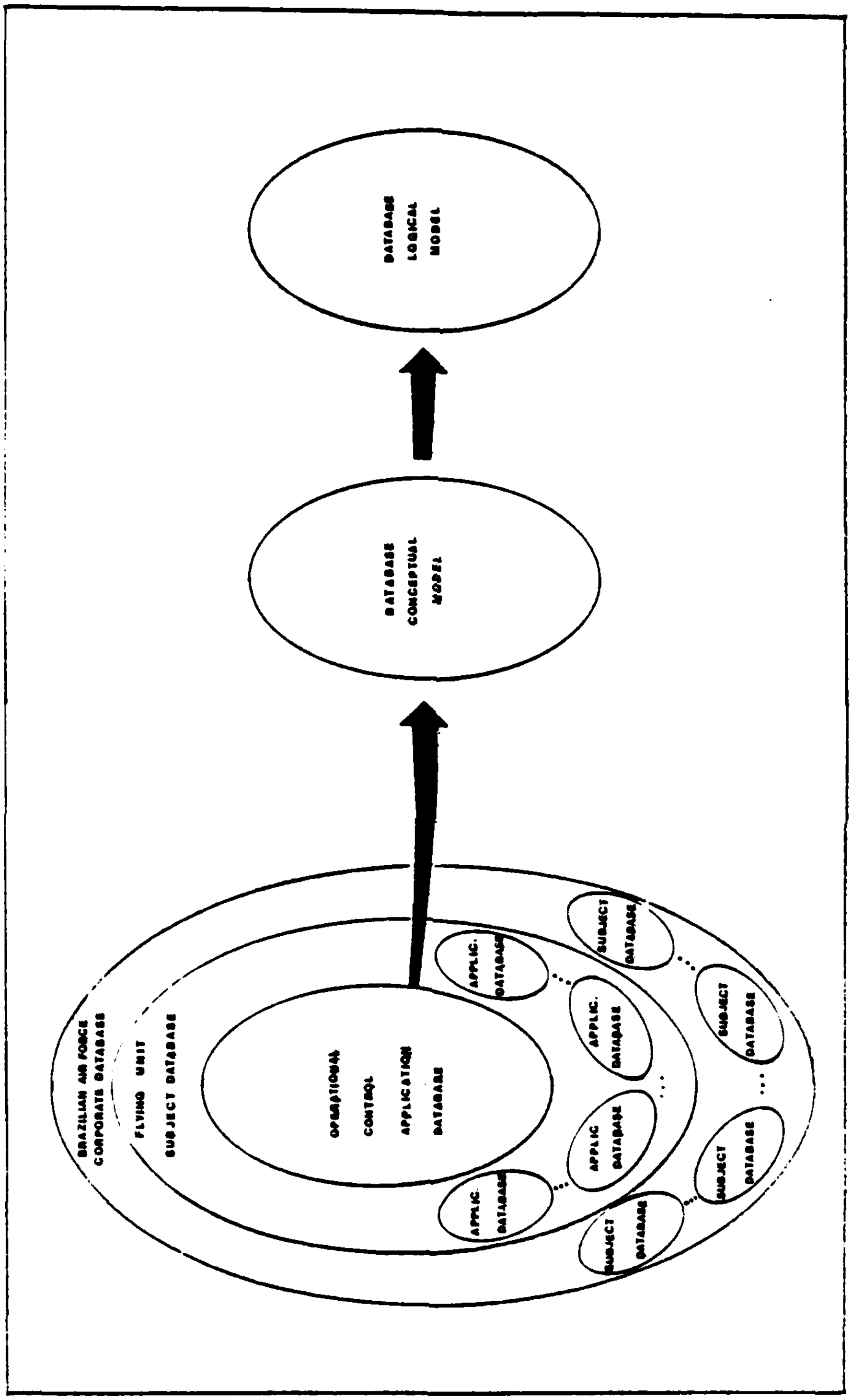


Figure 11 - Database Design Levels

In order to obtain the desirable system design modularization, a subject database ought to be composed of discrete modules of application databases, each of which is simple enough to be efficiently designed, completely understood by its design team, low in maintenance cost, and susceptible to high-productivity development methods, such as the use of high-level database languages. Application databases must fit into a stable and self-contained subject or functional database environment, and they will not do so unless designed with planning from the top.

A corporate wide planning of a database is vital, but corporate wide design of an integrated database is impratical. Instead, a bottom-up design of specific database application systems is needed. It is also feasible to do a top-down planning for data resources used by an application database system. Considered of strategic worth, top-down plannings must permit separate database design systems to be developed by themselves and must have the objective of achieving data consistency in all system levels. Each application database system design must be reasonably coherent, decoupled from other systems, small, not very complex to implement, and integrated by using data centrally defined (21).

The process of designing a database for the Brazilian Air Force Flying Unit Operations environment started almost as a natural continuation of the system analysis performed in chapter III. As shown in the next sections of this Chapter, a functional analysis was performed to determine the basic system functions and processes, then vital entities for the organization were identified, an entity relationship chart was derived, and finally potential database applications were identified to be developed next.

4.1.1 The Functional Analysis

As an organization composed of several different corporate system levels, the BAF primary mission is to provide aerospace forces capable of supporting the Nation's objectives in peace or war. To fulfill its mission, the BAF has been assigned primary interest in all operations in the Brazilian air space. In order to efficiently accomplish its primary mission, the BAF must effectively operate and control its flying units.

As one of the most important units in the BAF context, a flying unit is composed of a headquarters and two or more flights. Depending upon its specific assigned mission, a flying unit can be considered a squadron composed of two or more flights or a group composed of two or more squadrons. The flying unit is also considered the smallest air force unit operated separately.

Due to its very important and dynamic operational nature, a flying unit needs a very flexible, fast, and accurate data processing system, with some degree of intelligence, which can efficiently store, process, and retrieve information. This must be not only a file management system generating batch reports, but also an interactive system, able to help operating, controlling, and decision making processes.

A functional analysis is defined as a survey to determine basic functions in a specific system environment. It began with a survey and analysis of all identified main functions and processes that take place in a flying unit operations system environment (21).

To efficiently accomplish its assigned missions, a flying unit must perform some functions and processes. Function refers to the flying unit functional areas. Each functional area carries out a certain number of processes. Processes may be defined with simple definitions and should be basic activities and decision areas which are independent of any reporting hierarchy or specific management responsibility. To properly identify processes and functions, an environmental analysis was performed through all the stages in the life cicle of each type of product, service, and resource generated by a flying unit. During this functional analysis it was noticed that, the identification of functions and processes should be independent of the current organization chart, because the organization might change, but still have to carry out the same functions and processes (21). It was also noticed that functions and processes identified should represent fundamental concern for how the unit operates.

A thorough examination of relevant documents, organization charts, and regulations resulted in the following functions and associated processes.

1) Operational Planning - this function is composed of three basic processes:

a) analyzing previous operational plans - this process consists of collecting and analyzing data from previous "operational plans" of a "flying unit",

b) generating new operational plans - this process consists of generating an "operational plan" and projecting "mission orders" of a "flying unit",

c) controlling operational plans - this process consists of controlling the execution of current "operational plans" of a "flying unit".

2) The Operational Control - this function is composed of three basic processes:

a) planning the operations control - this process consists of planning the employment of available "aircraft" and "consumed items" in operational "flying unit" "missions" performed by qualified "crew members",

b) <u>generating mission orders</u> - this process is also called generating fragmented mission orders process and consists of generating a "written order" containing data about the "flying unit", "aircraft" to be employed, "item" to be consumed, "missions" to be performed, and qualified "crew members" to execute "missions",

c) <u>controlling the execution of operational mission</u> - this process consists of controlling the execution of "mission orders" in a "flying unit".

3) <u>Performing Mission</u> - this function is composed of the following three basic processes:

a) <u>planning the execution of mission</u> - this process consists of "crew member's" planning of the "mission" to be executed and "crew member's" preparing and pre-flying the "aircraft" to perform a "mission" in a "flying unit",

b) <u>employing aircraft</u> - this process consists of flying and employing the "aircraft" to accomplish "missions orders" in a "flying unit", c) <u>reporting executed mission</u> - this process consists of reporting data generated by the accomplishment of a "mission order" in a "flying unit".

4) <u>Training Crew members</u> - this function is composed of the following three basic processes:

a) <u>planning training</u> - this process consists of planning for "trainees" to be taught by "instructors" in specific "subjects", in order to became able to perform future "missions" in a "flying unit",

b) <u>providing training</u> - this process consists of providing "subjects" and "instructors" to teach "trainees" in accordance with the "flying unit" demand,

c) <u>evaluating training</u> - this process consists of controlling and evaluating "subjects", "trainees", and "instructors" of a "flying unit".

5) <u>Maintaining Aircraft</u> - this function is composed of the following three basic processes:

a) <u>planning maintenance</u> - this process consists of planning "maintenance personnel" and "maintenance material" to accomplish "maintenance services" and generate "aircraft" availability in a "flying unit",

b) <u>performing maintenance</u> - this process consists of employing "maintenance personnel and material" to perform "maintenance services" in order to make "aircraft" available to perform "mission orders" in a "flying unit",

c) <u>controlling maintenance</u> - this process consists of controlling and schedulling "maintenance services, personnel, and material" in a "flying unit".

6) <u>Supporting</u> - this function is composed of the following three basic processes of supporting personnel:

a) <u>supporting personnel</u> - this process consists of planning, controlling, and evaluating "supporting personnel" in order to support "flying unit" operations,

b) <u>supporting material</u> - this process consists of planning, employing, and controlling "supporting material" in order to support "flying unit" operations,

c) <u>supporting administration</u> - this process consists of planning, employing, and controlling "administrative supporting" in order to support "flying unit" operations.

7) Flying Safety - this function is composed of the following three basic processes:

a) preventing accidents - this process consists of planning, executing, and controlling of "accident prevention". It is performed by the "flight safety officer" in a "flying unit",

b) investigating accidents - this process consists of planning, executing and controlling "accident investigation". It is performed by the "flight safety officer" in a "flying unit",

c) controlling accident ocurrence - this process consists of planning, executing, and periodically evaluating "accident controls" in order to share data with "preventing accidents" and "investigating accidents processes".

These basic functions associated with a Flying Unit Operations System are shown in a pictorial form in Figure 12.

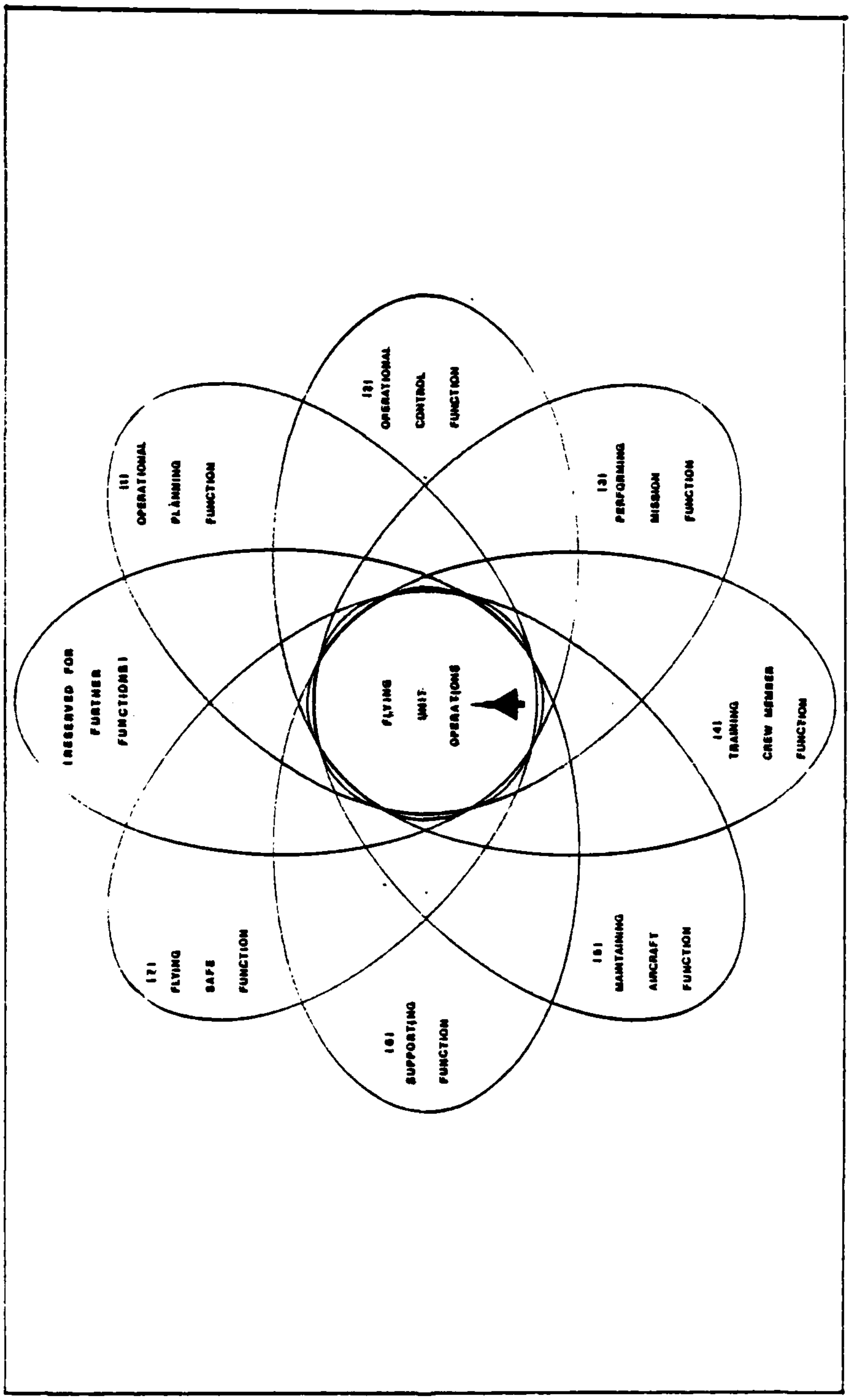


Figure 12 - Functions of a Flying Unit Operations System

4.1.2 System Vital Entities

The functional analysis described in the previous section was the basic source used to identify the flying unit operations system vital entities.

First of all, each function and process description that takes place in the operational environment was separately analyzed. Secondly, vital entities were identified. These entities are everything of interest for the system life, about which data had to be stored. Every key word used to describe functions and processes in the previous section, meaning a person, place, thing, object, or even some concept with characteristics of interest for a flying unit operations system, was identified as a system vital entity, by double quoting each key word occurrence (1). Finally, after eliminating redundancies, the following vital entities, in alphabetical order, were considered as an initial characterization of a flying unit operations system:

1) - accident;

2) - accident investigation;

3) - accident prevention;

4) - administrative supporting;

5) - aircraft;

6) - consumed items;

7) - crew members;

8) - flight safety officer;

9) - flying unit;

10) - instructors;

11) - maintenance personnel;

12) - maintenance material;

13) - maintenance services;

14) - mission;

15) - operational plan;

16) - subjects;

17) - supporting personnel;

18) - supporting material; and

19) - trainees.

4.1.3 <u>The Entity Relationship Chart</u>

After determining system vital entities in the previous section, they were analyzed to determine how they relate to each other. Afterwards, a pictorial format of their relationships were derived, by drawing the entity relationship chart shown in Figure 13.

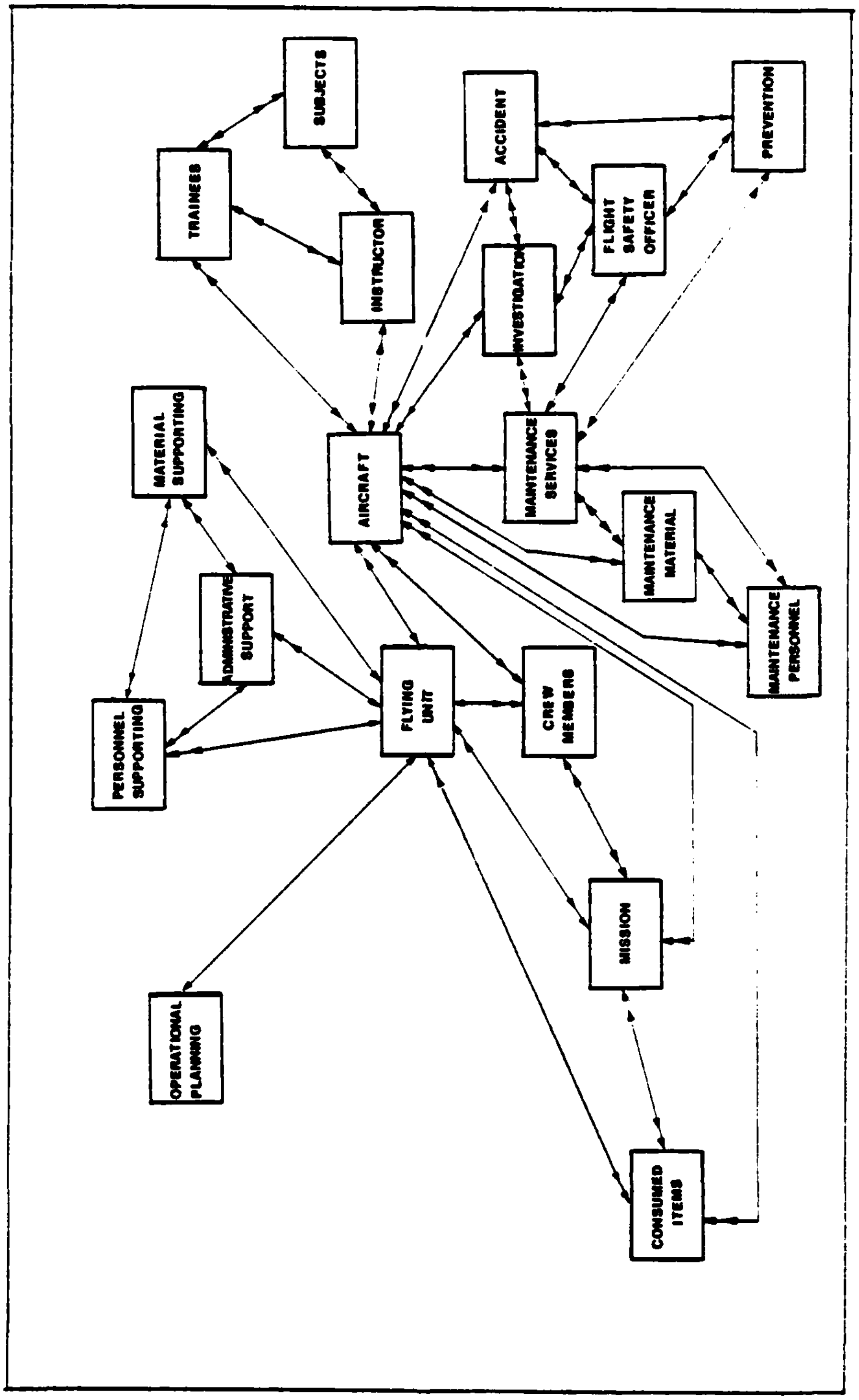


Figure 13 - The Entity Relationship Chart (21).

4.1.4 Potential Database Applications

Following the criteria to design the first application database system stated in Section 3.2.4, the first database to be developed should be one with a group of entities relatively fast and easy to implement. It also should be one that could solve immediate problems and have a fast payback.

The FLying UNIT Operational Control (FLUNITOC) database system shown in Figure 14 was selected to be developed first, because most of its entities and attributes could be directly derived from the current file management system. Its development could not only immediately solve the file management problems, but could also have the fastest payback by using all the existent supporting structure for the file mangement system. In addition to the Flying Unit Operational Control Database System, the subject developed in this thesis, it is clear from the entity relationship chart, that there are other potential database applications. These potential flying unit application databases are: 1) the flight safety database; 2) the maintenance database; 3) the operational planning database; 4) supporting database; and 5) training database.

Using data items defined during previous developments of application databases, it is desirable that potential flying unit application databases be implemented one at a time using commom defined data items and relations.

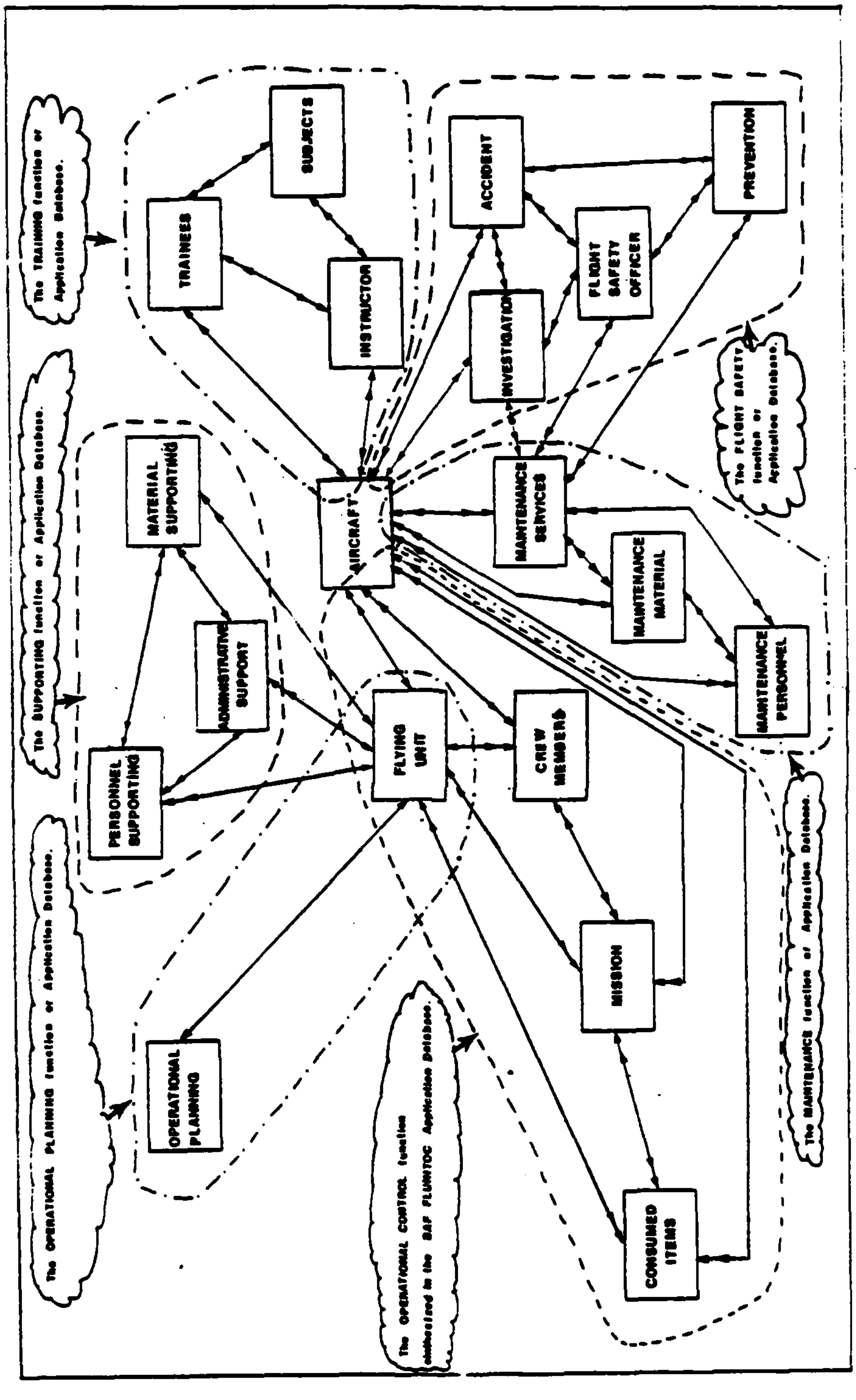


Figure 14 - Potential Database Applications

4.2 The Database Conceptual Model Design

In order to design a flying unit operational control database system, first the system requirements should be analyzed, then the users' views should be determined. To analyze system requirements the basic sources used were the current file management system reports. After analyzing the reports and eliminating all data item redundancies, each report was assumed as a user view. The users' views were considered inputs for designing the conceptual model, also known as a community view model.

The conceptual model refers to an inherent model of entities with the data items representing them, together with the relationships interconnecting the entities. The conceptual model gives an overall view of the flow of data in the system. It is designed and maintained by the Database Administrator (DBA) to help organize, visualize, plan and communicate the systems' relationships.

Afterwards, the conceptual model design is mapped to one of the three main database models, hierarchical, network, or relational, developing what is called the logical model (1).

4.2.1 Current File Management System Reports

The Brazilian Air Force has a number of flying units scattered among numerous Air Force Bases. A crew member may only operate in a specific flying unit and perform specific types of missions in specific types of aircraft. Each crew member uses a social security number (SSN) for identification. A crew member may perform several types of missions in a flying unit. In each mission, different items can be consumed or used in the aircraft. In the current file management system, reports are generated from these relationships to provide data for specific users (9).

Each file management system report was carefully analyzed as a new system requirement. Some repeated data were detected as appearing in two different reports delivered to the same user. Redundancies were eliminated and reports were simplified. It is sufficient that repeated data appear in just one report. Appendix A presents the description and depicts the layout of each report. The layout of Report #1, Individual Flight Record, is presented in Figure 15 to provide an example of the entire analytical process. The conceptual development of Report #1 is described in this section as an example of the process used for each report. The process was repeated for the 13 other reports.

Report #1 - Individual Flight Record

This report consists of all mission sorties performed by a given crew member during a specific period. Every flying unit may have several different crew members at a certain point in time. Report #1 is printed for each crew member on a monthly or periodic basis.

```
*****************************************************************
* BRAZILIAN AIR FORCE                    R_DATE : 02/01/85      *
* FLYING UNIT OPERATIONAL CONTROL SYS   (Report Date)           *
* F_UNITNO: 2732-1353                    R_TIME : 00:00:00      *
* (Flying Unit Number)                   (Report Time)          *
* R_NO : 01                              R_SDATE: 01/01/85      *
* (Report Number)                        (Report Start Date)    *
* R_NAME: INDIVIDUAL FLIGHT RECORD       R_EDATE: 01/31/85      *
* (Report Name)                          (Report End Date)      *
*                                                               *
* (Crew Member Social Sec. No.) C_SSN   : 291-80-1970          *
* (Crew Rank, Speciality) C_RANK, C_SPEC: Major, Pilot         *
* (Crew Last Name) C_LNAME              : Cunha                *
* (Crew Complement Name) C_CNAME        : Adilson M. da        *
*---------------------------------------------------------------*
* S_DEPDAT      S_CODENO      S_MTCODE      A_TYPE     A_NO    *
* (Sortie       (Sortie       (Sortie       (Acft      (Acft   *
* departure      Code         Mission       Type)      No.)    *
* date )         number)      Type Code)                       *
*                                                               *
* 01/01/85      10408701       07FM03       F-103B      2120   *
* 01/01/85      10408702       14TG00       F-103B      2120   *
* 01/02/85      10408901       07FM03       F-103E      2123   *
* --/--/--      --------       ------       ------      ----   *
*   ...           ....           ..          ....        ..    *
*---------------------------------------------------------------*
*S_XFUNCT   S_DEPLOC    S_ARRLOC   S_TIME   S_LANDNO  S_ICOND  *
*(Sortie    (Sortie     (Sortie    (sortie  (Sortie   (Sortie  *
*executed   departure   arrival    time)    landing   instrum.*
*function)  location)   location)           number)   condit.)*
*                                                               *
*   1P        SBRJ        SBBR       1.5       02        R      *
*   IN        SBBR        SBRJ       1.6       03        R      *
*   IN        SBRJ        SBRJ       0.8       04        S      *
*   --        ----        ----       -.-       --        -      *
*   ..         ..          ..         .        ..        .      *
*---------------------------------------------------------------*
* S_IFUNCT    S_IFTIME    S_IDPCND      S_IDPLOC     S_IDPNO   *
* (Sortie     (Sortie     (Sortie       (Sortie      (Sortie   *
* instrum.    instrum.    instrum.      instrum.     instrum   *
* function)   function    descend       descend      descend   *
*             time)       procedure     procedure    proced.   *
*                         condition)    location)    number)   *
*                                                               *
*    1P          0.4          R            SBBR         02     *
*    IN          0.6          R            SBRJ         01     *
*    IN          0.7          S            SBRJ         03     *
*    --          -.-          -            ----         --     *
*    ..           .           .             ..          ..     *
*****************************************************************
```

Figure 15 - The Report # 1 Individual Flight Record Layout

The Report # 1 Individual Flight Record and the remaining thirteen analyzed reports represent information needs of the current file management system. They are described in Appendix A. All fourteen reports in pictorial format are shown in Figure 16.

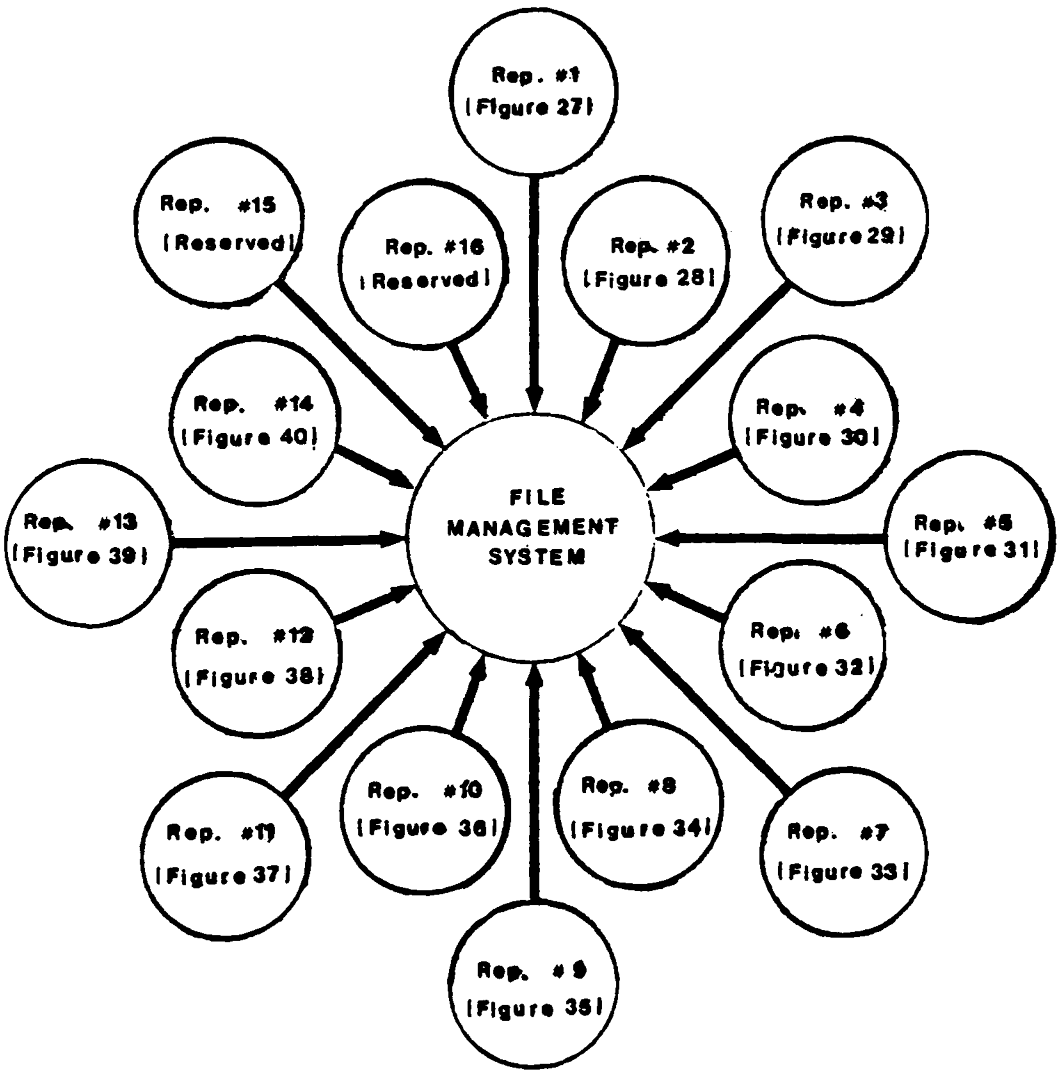


Figure 16 - Current File Management System Reports

4.2.2 The System Data Dictionary

After analysing each report individually, a data dictionary was constructed. This data dictionary was planned to be a central repository of information about the entities: the data items representing the entities, the relationships between the entities, their origins, meanings, uses, and representation formats (1).

The database expands as applications are developed and integrated. New data items are introduced, and data items used for the database design may have to be modified. The system data dictionary intends to be a facility that provides uniform and central information about all the data resources. But, considering the main purpose of this thesis and mainly due to time constraints, the system data dictionary initially contains only a collection of data items definitions. Each data item was identified by a unique code name and described when referred to in the file management system reports, representing the entities and relationships between entities. The system data dictionary is shown in the seven pages of Appendix B, in alphabetical order.

4.2.3 Report Assumptions

After studying and analysing each report as a system requirement, several assumptions were made in order to constrain the system behavior. These assumptions described below were not stated to make an easier solution for the data processing problems, but rather to reflect an actual Brazilian Air Force Flying Unit Operations System in a database system design. Each data item can be decodified in Appendix B.

1) Flying unit numbers (F_UNITNO) are unique.

2) At a given point in time, a flying unit number (F_UNITNO) can have several different aircraft types (A_TYPE), aircraft numbers (A_NO), crew members (C_SSN), can perform several sortie mission types (S_MTCODE), and consume several items type codes (S_ITCODE).

3) The crew member social security number (C_SSN) is unique.

4) The aircraft number (A_NO) is unique.

5) Sortie code numbers (S_CODENO) are unique.

6) Sortie mission type codes (S_MTCODE) are unique.

7) Sortie item codes (S_ITCODE) are unique.

8) A sortie item code (S_ITCODE) uniquely identifies a sortie item name (S_ITNAME).

9) A sortie item code (S_ITCODE) with a sortie item recept number (S_ITRCNO) uniquely identify a sortie item consumed quantity (S_ITCOQY) and a sortie item supplier (S_ITSUPP).

10) Report numbers (R_NO) are unique.

11) A report number (R_NO) uniquely identifies a

report name (R_NAME).

12) A report number (R_NO) with report date (R_DATE) and report time (R_TIME) uniquely identify a report that has a starting date (R_SDATE) and ending date (R_EDATE).

13) A Flying unit code number (F_UNITNO) is uniquely identified by a sequence of four pairs of numbers. The first pair identifies a numbered air force number (B_NUMBAF), the second pair identifies a wing number (B_WING), the third pair identifies a group number (B_GROUP), and the last pair identifies a squadron number (B_SQDR) uniquely identify.

14) At a given point in time, a crew member (C_SSN) can be assigned to only one flying unit (F_UNITNO).

15) A sortie code number (S_CODENO) is composed of a mission order number and a sortie number and it is considered the smallest part of a mission to be reported.

16) A sortie code number (S_CODENO) uniquely identifies a sortie mission type code (S_MTCODE).

17) In a given administrative unit (C_ADUNIT) there is a number of crew members who fly certain types of aircraft (A_TYPE) for certain flying unit number (F_UNITNO).

18) At a given point in time, a crew social security number (C_SSN) uniquely identifies a crew member rank (C_RANK), a crew speciality (C_SPEC), a crew last name (C_LNAME), a crew complement name (C_CNAME), a crew functional qualification code (C_FQCODE), and a crew administrative unit (C_ADUNIT).

19) An aircraft number (A_NO) uniquely identifies an aircraft type (A_TYPE), a total cell time aircraft number (T_CELANO) and a total cell landing aircraft number (T_CELALN).

20) At different points in time, an aircraft number (A_NO) can have different numbers of periods status, such as: aircraft number of periods on status A (A_NPSTAA), number of periods on status B (A_NPSTAB), number of periods on status C (A_NPSTAC), number of periods on status D (A_NPSTAD), number of periods on status E (A_NPSTAE), number of periods on status F (A_NPSTAF), number of periods on status G (A_NPSTAG), and number of periods available (A_NPAVAL).

21) A crew member functional qualification code (C_FQCODE) uniquely identifies a sortie executed function (S_XFUNCT).

22) At a given point in time, a flying unit number

(F_UNITNO) has only one operational controller, one material controller, and one personnel controller.

23) At different points in time, a crew member (C_SSN) can perform several different types of missions (S_MTCODE) in several different aircraft types (A_TYPE) and numbers (A_NO).

24) At different points in time, an aircraft number (A_NO) can perform several different types of missions (S_MTCODE), consume different items quantities (S_ITCOQY) of sortie items type codes (S_ITCODE).

25) At a given point in time, an aircraft number (A_NO) can perform missions only for one flying unit (F_UNITNO).

26) At a given point in time, a crew member (C_SSN) can perform missions (S_MTCODE) only for one flying unit (F_UNITNO).

4.2.4 Third Normal Form (3NF) Relations

To develop the third normal form relations for each report, the set of data items within each report was identified. Then the relationships between data items were determined and described by identifying the key data items and nonkey data items for each relation. Finally, the third normal form relations for each set of data items was derived. Where this was not possible for individual reports, the data from reports were merged to establish the third normal form relations (1).

The normalization process described in Chapter III is applied for each of the fourteen system reports in Appendix C. Report #1 is used in this section as an example of this process.

Report #1 - Individual Flight Record

a) The data items representing the entities of this report are: F-UNITNO, R-NO, R-NAME, C-SSN, C-RANK, C-SPEC, C-LNAME, C-CNAME, S-DEPDAT, S-CODENO, S-MTCODE, A-TYPE, A-NO, S-XFUNCT, S-DEPLOC, S-ARRLOC, S-TIME, S-LANDNO, S-ICOND, S-IFUNCT, S-IFTIME, S-IDPCND, S-IDPLOC, and S-IDPNO.

b) The relationships between the data items of this report are:

(1) R-NO <----------> R-NAME, that means, for a given report number (R-NO) there is only one report name (R-NAME), that is, a one-to-one mapping represented as <-------> two opposite arrows;

(2) R-NO,R-DATE,R-TIME <----------> R-SDATE, R-EDATE, that means, for a given report number (R-NO) with report date (R-DATE) and report time (R-TIME) there is only one report starting date (R-SDATE) and report ending date (R-EDATE), that

is, a one-to-one mapping;

(3) F-UNITNO <--<-------> C-SSN, A-NO, that means, for a given flying unit number (F-UNITNO), there may be many crew members social security numbers (C_SSN) and aircraft numbers (A-NO), that is, a one-to-many mapping, represented as <--<------>;

(4) S-CODENO <--<-------> S-MTCODE, that is, for a given sortie code number (S-CODENO), there may be many sortie mission type code (S-MTCODE);

(5) F-UNITNO,S-CODENO <--<-------> S-MTCODE, S-DEPLOC, S-ARRLOC, S-TIME, S-LANDNO, A-NO, C-SSN, that is, for a given flying unit number (F_UNITNO) with sortie code number (S-CODENO) there may be many sortie mission type code (S-MTCODE), sortie departure location (S-DEPLOC), sortie arrival location (S-ARRLOC), sortie time (S-TIME), sortie landing number (S-LANDNO), aircraft number (A-NO), and crew member social security number (C-SSN);

(6) C-SSN <--<-------> C-RANK, C-SPEC, C-LNAME, C-CNAME, C-FQCODE, that is, for a given crew member social security number (C-SSN) there may be many crew member rank (C-RANK), crew speciality (C-SPEC), crew last name (C-LNAMF', crew complement name (C-CNAME), and crew functional qualification code (C-FQCODE);

(7) C-FQCODE <--<-------> S-XFUNCT, that is, for a

given crew member functional qualification code (C-FQCODE) there may be many crew members executed functions (C-XFUNCT);

(8) A-NO <--<-------> A-TYPE, that is, for a given aircraft number (A-NO) there may be many aircraft type (A-TYPE); and

(9) S-CODENO,S-DEPDAT,A-ANO <--<-------> C-SSN, S-XFUNC, S-IFUNCT, S-IFTIM, S-IFCOND, S-IDPLOC, S-IDPCON, S-IDPNO, that is, for a give sortie code number (S-CODENO) with sortie departure date (S-DEPDAT) and also with aircraft number (A-NO) there may be many crew member social security number (C-SSN), sortie executed function (S-XFUNCT), sortie instrument function (S-IFUNCT), sortie instrument function time (S-IFTIM), sortie instrument function condition (S-IFCOND), sortie instrument descend procedure location (S-IDPLOC), sortie instrument descend procedure condition (S-IDPCON), and sortie instrument descend procedure number (S-IDPNO).

c) The third normal form relations for the Individual Flight Record Report are:

(1) R-NO <----------> R-NAME;

(2) R-NO,R-DATE,R-TIME <----------> R-SDATE, R-EDATE;

(3) F-UNITNO <--<-------> C-SSN, A-NO;

(4) S-CODENO <--<--------> S-MTCODE;

(5) F-UNITNO,S-CODENO <--<--------> S-DEPLOC, S-ARRLOC, S-TIME, S-LANDNO;

(6) C-SSN <--<--------> C-RANK, C-SPEC, C-LNAME, C-CNAME, C-FQCODE;

(7) C-FQCODE <--<--------> S-XFUNCT;

(8) A-NO <--<--------> A-TYPE; and

(9) S-CODENO,S-DEPDAT,A-ANO <--<--------> C-SSN, S-XFUNCT, S-IFUNCT, S-IFTIM, S-IFCOND, S-IDPLOC, S-IDPCON, S-IDPNO.

This process was repeated for the remaining thirteen reports in order to determine all relations needed to generate each report. The 3NF for the first report and the remaining thirteen are shown in Appendix C, The BAF FLUNITOC Reports 3NF Relations.

4.2.5 Summary of Third Normal Form (3NF) Relations

To summarize and level third normal form (3NF) relations, first, all the relations from all the reports were examined and the redundant relations were rejected. Then the third normal form relations were checked, and after that, remaining relations were leveled per number of key data items and numbered within levels (1).

First Level of Third Normal Form Relations

This level is also known as the entity level, because each relation that contains just one data item as its primary key may be considered as a system entity. The first level contains the following relations with just one data item as primary key:

101)	R-NO	<----------->	R-NAME;
102)	F-UNITNO	<--<--------->	C-SSN, A-NO;
103)	C-SSN	<----------->	C-RANK, C-SPEC, C-LNAME, C-CNAME, C-FQCODE, C-ADUNIT, T-CF1TIM, T-CF1LND, T-CF2TIM, T-CF2LND, T-CF3TIM, T-CF3LND, G-TLNDNO, G-TCSTIM;
104)	C-FQCODE	<--<--------->	S-XFUNCT;
105)	A-NO	<--<--------->	A-TYPE, T-CELANO, T-CELALN;
106)	S-CODENO	<--<--------->	S-MTCODE;
107)	S-ITCONO	<----------->	S-ITNAME, S-ITRCNO;
108)	S-ITRCNO	<--<--------->	S-ITCOQY, S-ITSUPP.

Second Level of Third Normal Form Relations

The second level contains the following relations with two data items as primary keys:

201) F-UNITNO,S-MTCODE <-----------> G-TMSTIM, S-DEPLOC, S-ARRLOC, S-TIME, S-LANDNO;

202) A-NO,D-YEAR <--<--------> T-YANO, T-YALNDN;

203) C-SSN,D-YEAR <-----------> T-YCSTIM.

Third Level of Third Normal Form Relations

The third level contains the following relations with three data items as primary keys:

301) R-NO,R-DATE,R-TIME <-----------> R-SDATE, R-EDATE;

302) F-UNITNO,A-TYPE,C-SSN <-----------> G-TCSTIM, T-PATYPE, T-SQCTIM;

303) A-NO,S-MTCODE,D-YEAR <-----------> T-YMATIM, T-YMTANO;

304) F-UNITNO,A-TYPE,C-ADUNIT <-----------> AD-UCTOT;

305) A-TYPE,C-ADUNIT,D-YEAR <-----------> AD-UYTIM;

306) A-NO,R-SDATE,R-EDATE <--<--------> T-PMSTIM, T-PALNDN, A-NPSITA, A-NPSITB, A-NPSITC, A-NPSITD, A-NPSITE, A-NPSITF, A-NPSITG, A-NPAVAL, S-ITNAME, T-PICOQY, S-ITUNIT;

307) C-SSN,R-SDATE,R-EDATE <----------> T-PCXFNC,
T-PMSTIM.

Fourth Level of Third Normal Form Relations

The fourth level contains the following relations with four data items as primary keys:

401) F-UNITNO,S-MTCODE,R-SDATE,R-EDATE <-<--> T-PMSTIM;

402) F-UNITNO,A-TYPE,C-SSN,D-YEAR <----> T-YCSTIM;

403) A-TYPE,C-ADUNIT,R-SDATE,R-EDATE <----> AD-UPTIM;

404) A-NO,S-MTCODE,R-SDATE,R-EDATE <----> S-TIME,
S-ITCONO,
S-ITCOQY,
S-DEPLOC,
S-ARRLOC,
T-PMANO,
T-TPMANO,
T-MSTIME,
T-PICOQY;

405) S-CODENO,A-NO,S-DEPDAT,S-DEPTIM <-<--> C-SSN,
S-XFUNCT,
S-IFUNCT,
S-IFTIM,
S-IFCOND,
S-IDPLOC,
S-IDPCON,
S-IDPNO,
S-ITRCNO,
S-DEPLOC.

Fifth Level of Third Normal Form Relation

The fifth level contains the following relation with five data items as primary keys:

501) F-UNITNO,A-TYPE,C-SSN,R-SDATE,R-EDATE <----> T-PCREW.

4.2.6 Deriving the Conceptual Model

To derive the conceptual model, all the normalized relations from the previous section were drawn in a diagramatical form, as shown in Figure 17. The relations containing only one data item were named entities and placed on the first level, also called the entity level.

According to the number of key data items, each relation was placed into a corresponding level and linked to one or more entities, depending on its number of key data items (1).

If no corresponding entity was found on the first level, the relation was re-analyzed to make sure of its correctness. After its refinement or confirmation, if the relation still had as a key, a data item with no entity correspondence, a new entity had to be created on the first level to provide the desirable link to derive the complete conceptual model.

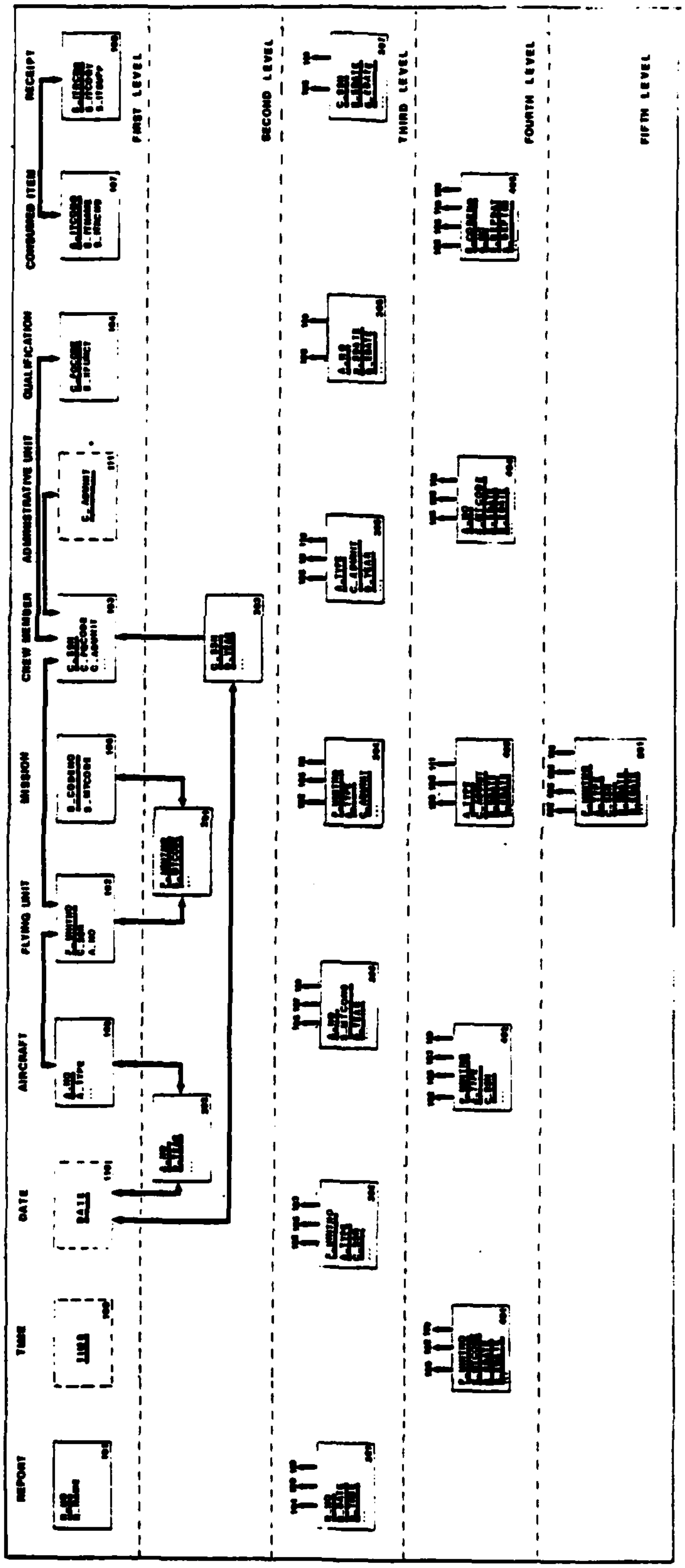


Figure 17 - The Database Conceptual Model (1).

4.3 The Database Logical Model Design

The version of the conceptual model that can be presented to a database mangement system (DBMS) is called a logical model. The users are presented with subsets of this logical model. These subsets, also referred to as subschemas in the literature, are called external models. The external models are the views that the users get based on the logical model. This terminology such as "internal model," "conceptual model," and "external model" is from American National Standards Institute (ANSI/X3/SPARC) Database Management System's Study Group.

The logical model is mapped to physical storage such as disk, tape, etc. The Physical model, which takes into consideration the distribuition of data, access methods, and indexing techniques, is called an internal model. The logical model can be either a relational, a hierarchical, or a network data model. The term data model here is used in the generic sense. It may be applied to a conceptual or logical or internal (physical) model. The DBMS is not a factor to be considered in designing a conceptual model, but designing a logical model is dependent on the DBMS to be used (1) (20).

In developing a logical model of the database, the relational data model was chosen as the best one for the conceptual model designed in Section 4.2. Mapping the conceptual model to a relational data model consists of defining relations and attributes. The relational data model consists of a number of relations that can be represented in the form of tables, as shown in the next section.

4.3.1 Selecting Relations to Implement

The following sample of relations were selected from section 4.2.5, in order to use the layout of Report #7 shown in Figure 18 as an example to support the Report Generator application program implementation in Chapter V. Besides its number, each relation received a name which identifies the correspondent table and database implementation. This sample of implemented database relations are shown in Appendix E using INGRES DBMS.

101) R-NO <------------> R-NAME;

102) F-UNITNO <--<---------> C-SSN, A-NO;

103) C-SSN <------------> C-RANK, C-SPEC, C-LNAME, C-CNAME, C-FQCODE, C-ADUNIT, T-CF1TIM, T-CF1LND, T-CF2TIM, T-CF2LND,T-CF3TIM, T-CF3LND, G-TLNDNO, G-TCSTIM;

105) A-NO <--<---------> A-TYPE, T-CELANO, T-CELALN;

301) R-NO,R-DATE,R-TIME <-----------> R-SDATE, R-EDATE;

404) A-NO,S-MTCODE,R-SDATE,R-EDATE <----> S-TIME, S-ITCODE, S-ITCOQY, S-DEPLOC, S-ARRLOC, T-PMANO, T-TPMANO, T-MSTIME, T-PICOQY;

```
**************************************************************
*                                                            *
* BRAZILIAN AIR FORCE                    R_DATE : 02/01/85   *
*                                        (Report Date)       *
* FLYING UNIT OPERATIONAL CONTROL SYS    R_TIME : 00:00:00   *
*                                        (Report Time)       *
* F_UNITNO: 2732-1353                    R_SDATE: 01/01/85   *
* (Flying Unit Number)                   (Report start date) *
*                                        R_EDATE: 01/31/85   *
* (Report Number) R_NO: 07               (Report end date)   *
*                                                            *
* R_NAME: ITEMS PER MISSION                                  *
* (Report Name)                                              *
*                                                            *
*------------------------------------------------------------*
*   S_MTCODE        A_TYPE        A_NO           S_DEPLOC    *
*   (Sortie        (Aircraft     (Aircraft       (Sortie     *
*    Mission        Type)         Number)        Departure   *
*    Type                                        Location)   *
*    Code)                                                   *
*                                                            *
*    07FM03         F-103B         2120            SBRJ      *
*    07FM03         F-103E         2120            SBBR      *
*    07TG00         F-105F         2123            SBRJ      *
*    ------         ------         ----            ----      *
*      ..             ..            ..              ..       *
*                                                            *
*------------------------------------------------------------*
*    S_ARRLOC       S_TIME       S_ITCODE      S_ITCOQY      *
*    (Sortie       (Sortie       (Sortie       (Sortie       *
*    Arrival        Time)        Item          Item          *
*    Location)                   Code)         Consumed      *
*                                              Quantity)     *
*                                                            *
*     SBBR          1.5           MISS77         02          *
*     SBRJ          1.6           ROCK26         04          *
*     SBRJ          0.8           BOMB15         08          *
*     ----          --.-          ------         --          *
*      ..            .              ..           ..          *
*                                                            *
*------------------------------------------------------------*
*Periodic Total Time per Mission Code Type T_MSTIME: 213.1*
*                                          --------  ---.-*
*                                            ....     ... *
*                                                            *
*Periodic Total per Item Consumed Quantity T_PICOQY:    14 *
*                                          --------     -- *
*                                            ....       .. *
*                                                            *
**************************************************************
```

Figure 18   The Report #7 Items per Mission Layout

4.3.2 Sample Tables for Selected Relations

The six relations chosen from Section 4.2.5 are necessary to generate a specific user view represented by Report #7. In this case, the term user view refers to an external view, that is, the totality of data seen by the user of Report #7. This specific user view is considered a fraction of the database logical model composed of the six relations and derived from the conceptual model in Figure 17. The redundant key attributes that appeared in more than one user views were eliminated after each relation was studied separately, as shown in Appendix C. However, the relations below, representing a specific user view, are not necessarily implemented physically in that way.

The mapping process of the conceptual model onto a relational data model is performed using table formats. Every box from the conceptual model becomes a relation or a table that can supply one or more user views. This process becomes relatively easy when a relational approach was also used to design the conceptual model.

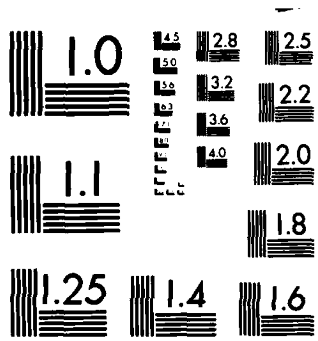
A fraction of the logical model represented by sample tables for the six selected relations are presented as follows.

The REPORTID Relation and sample Table

101) R-NO <---------> R-NAME.

AD-A151 848 A DATABASE DESIGN FOR THE BRAZILIAN AIR FORCE FLYING UNIT OPERATIONAL CON..(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. A M DA CUNHA 2/3

UNCLASSIFIED 14 DEC 84 AFIT/GCS/ENG/84D-7 F/G 9/2 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963 A

R-NO	R-NAME
01	Individual Flight Record
02	Mission Type Summary
04	Aircraft Numbers Missions Summary
07	Items per Missions

Table 1 - The Reportid Table

The FLYUNIT Relation and Sample Table

102) F-UNITNO <--<--------> C-SSN, A-NO.

F-UNITNO	C-SSN	A-NO
02132732	291801970	2120
02132732	291801970	2220
02132732	391801970	2121
02132733	491801923	2122
03110234	591811724	2223
03110234	691801613	2301

Table 2 - The Flyunit Table

The CREW Relation and Sample Table

103) C-SSN <-----------> C-RANK, C-SPEC, C-LNAME, C-CNAME, C-FQCODE, C-ADUNIT, T-CF1TIM, T-CF1LND, T-CF2TIM, T-CF2LND,T-CF3TIM, T-CF3LND, G-TLNDNO, G-TCSTIM.

C-SSN	C-RANK	C-SPEC	C-LNAME	C-CNAME	C-FQCODE
391801970	Capt	Pilot	John		FP
291801970	Major	Pilot	Cunha		IP
691801613	LtCol	Naviga	Foster		FN
591811724	Capt	Fl Eng	David		FF
491801923	Capt	Pilot	Ali		IP

Table 3 - The Crew Table

The AIRCRAFT Relation and Sample Table

105) A-NO <--<--------> A-TYPE, T-CELANO, T-CELALN.

A-NO	A-TYPE
2120	MIRAGE
2121	MIRAGE
2122	MIRAGE
2220	F-16
2223	F-16
2301	F-15

Table 4 - The Aircraft Table

The REPORT Relation and Sample Table

301) R-NO,R-DATE,R-TIME <----------> R-SDATE, R-EDATE.

R-NO	R-DATE	R-TIME	R-SDATE	R-EDATE
01	841216	100000	841201	841215
02	841216	100500	841201	841215
04	841216	101000	841201	841215
07	841216	120000	841201	841215
07	850116	000000	850101	850115

Table 5 - The Report Table

The MISTYPE Relation and Sample Table

404) A-NO,S-MTCODE,R-SDATE,R-EDATE <----> S-TIME, S-ITCODE, S-ITCOQY, S-DEPLOC, S-ARRLOC, T-PMANO, T-TPMANO, T-MSTIME, T-PICOQY;

A-NO	S-MTCODE	R-SDATE	R-EDATE	S-TIME	S-ITCODE
2121	Intercept			2.7	Miss77
2121	Intercept			1.3	Miss24
2120	Combat			0.7	Gunshot
2220	Grnd Supp			2.1	Gunshot
2223	Formation			1.4	Rocket1
2301	Formation			1.5	Rocket3

Table 6 - The Mistype Table

V. System Implementation

The total system implementation is not the purpose of this research because it requires effort that is beyond the scope of a thesis. In this chapter the system was partially implemented at three different levels.

On the first level of implementation, the Decision Support System concepts in the overall system environment were applied. On the second level of implementation, a database system menu and a report generator application program were developed. This level was implemented in order to complete the database system design process. Taking advantage of the relational database structure, database query retrieval was considered on the third level of implementation. Representing an efficient way to retrieve information, it was recognized as another system requirement specification feasible to implement in this thesis.

5.1 The First Level of Implementation - DSS

For this level a Dialog Generator Management Software (DGMSW) Prototype was projected and implemented. It represents the way the author sees an overall system implementation applying the DSS theory to solve decision making problems in all levels of the system environment.

5.1.1 Projecting a DGMSW Prototype

Keeping in mind the importance of the environmental system analysis performed in Chapter III, a menu-driven program was projected to show how a decomposed system and sub-systems could be implemented through Dialog Generator Management Software (DGMSW) using the Decision Support System theory.

This system was projected in order not only to better understand the overall system environment but also to determine how different levels of systems could be related to each other and interactively respond and integrate all needs of DSS and the decision making process. A menu dialog was chosen because it seems to be quite effective for inexperienced or infrequent users who are familiar with the problem to be solved. For DSS to provide a large number of functions, menu dialogs often require many menu items, and in such cases the menus should be structured. A well-designed dialog generator does not guarantee the success of DSS implementations, but it is considered a necessary ingredient (26). Figure 19 shows the main structure adopted for a sample Dialog Generator Management Software following the DSS theory stated in Chapter III and the DSS architecture model approach from Figure 3.

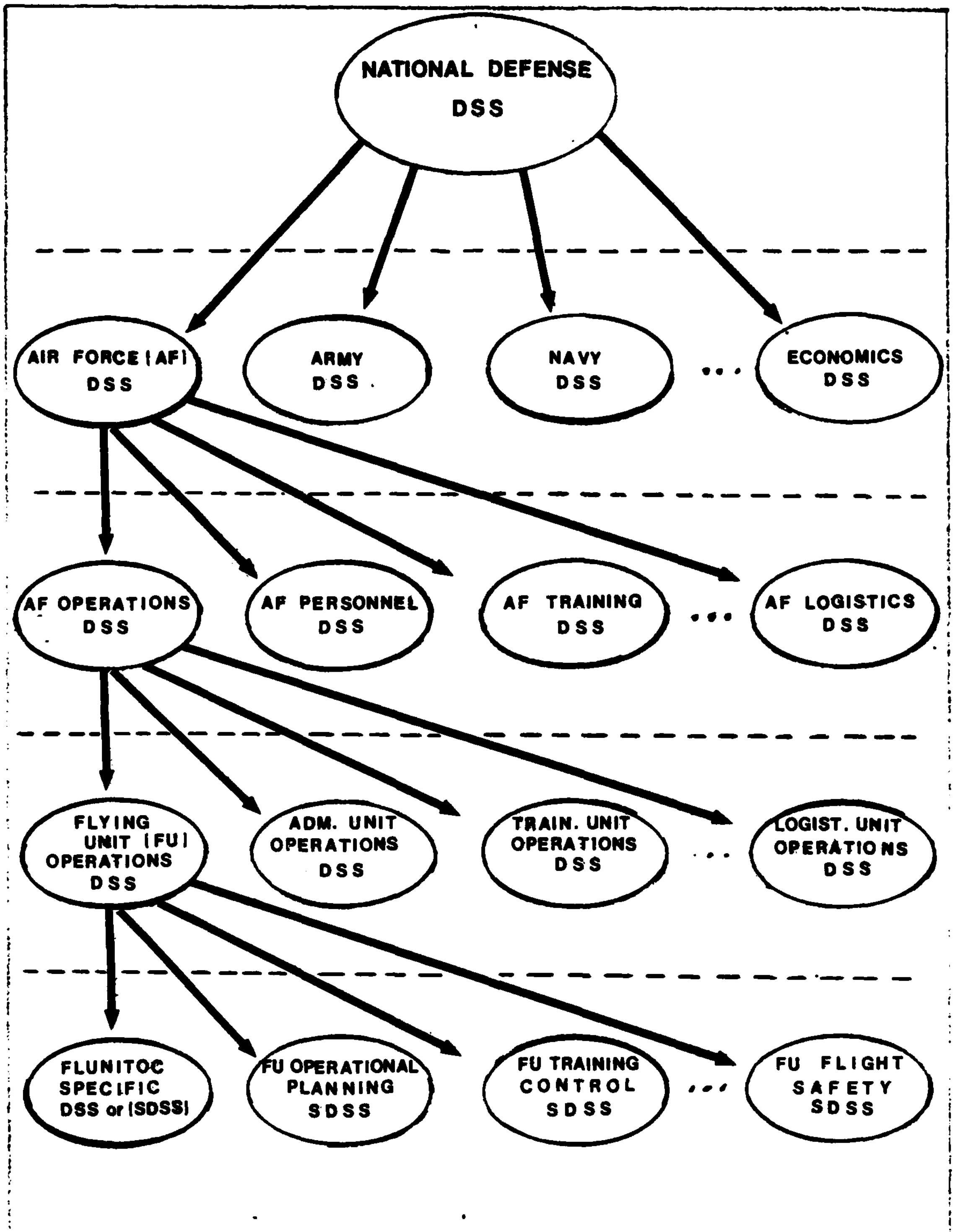


Figure 19 - A Dialog Generator Management Software Prototype

5.1.2 Implementing a DGMSW Prototype

The structure of the Dialog Generator Management Software Prototype was implemented through several menu-driven programs using dBASE II DBMS running on a Z-80 microcomputer. Appendix D details this first level of implementation showing the created menus and the implemented programs (27).

5.2 The Second Level of Implementation - Database

In order to accomplish the database system design, a partial database system was implemented in this second level. This partial implementation was performed by following the six different steps described in this section: projecting database system implementation, choosing database management system (DBMS), mapping selected tables to INGRES DBMS, using INGRES data manipulation language (DML), selecting database access method structures, and implementing the system menu and a sample report generator.

5.2.1 Projecting Database Implementation

A Flying Unit Operational Control System main menu was projected in this section as shown in Figure 20. In order to fulfill all system needs six options were projected: data entry, report generation, update transaction, query transactions, user help, and quit. It was assumed that most of the data entry proceedings from the current file management system could be converted to the database application system, using the interactive on-line features of a DBMS without major problems. Consequently, this issue was not considered for this thesis work. Report generation proceedings could be implemented through application programs designed for each user view or report need (15) (16) (34).

As an example of this implementation process, Report #7 Items per Missions, shown in Figure 16, was considered to be later implemented through an application program. Three basic types of update transactions were projected consisting of data item insertions, deletions, and modifications. Update transactions could be performed interactively from a terminal through three basic application programs to be developed one for each type of update. But this issue was not considered for this thesis work. Query transactions consisting of "on line" database transactions were considered among the most effective database retrieval techniques that allow quick and efficient information retrieval from databases. Mainly due to its importance, dynamic performance, and fundamental human-thinking nature, this issue was addressed separately in Section 5.3.

```
****************************************************************
*                                                              *
* BRAZILIAN AIR FORCE                                          *
*                                                              *
* FLYING UNIT OPERATIONAL CONTROL SYSTEM                       *
*                                                              *
* ***  MAIN MENU  ***                                          *
*                                                              *
*--------------------------------------------------------------*
*                                                              *
*      OPTIONS:                                                *
*                                                              *
*         1      DATA ENTRY                                    *
*         2      REPORT GENERATION                             *
*         3      UPDATE TRANSACTION                            *
*         4      QUERY TRANSACTION                             *
*         5      HELP                                          *
*         6      QUIT                                          *
*                                                              *
*      SELECT OPTION -->                                       *
*                                                              *
****************************************************************
```

Figure 20 - The FLUNITOC System Main Menu Layout

5.2.2 Choosing the Database Management Systems (DBMS)

As stated before in this chapter, the dBASE II DBMS was used to implement the overall system conceptualization of the analytical part of this thesis applying the Decision Support System theory. The database design part of the Flying Unit Operational Control System was chosen to be implemented on the AFIT VAX 11/780 computer. The INGRES DBMS runing under the UNIX time sharing system was the available system to use.

5.2.3 Mapping Tables to INGRES DBMS

The procedure of creating a database using INGRES had to be performed only once. First of all, an empty database named FLUNITOC, meaning FLying UNIT Operational Control System, was generated. Once the empty FLUNITOC database was generated, each relation had to be defined using the INGRES data description language (DDL), as shown in Appendix E. This was performed by mapping each selected table with its attribute names, dimensions, and characteristics, designed in Section 4.3.2, into INGRES DBMS (34). After that, each of the six initially selected tables filled with the sample data from Section 4.3.2 was loaded into separate UNIX files and afterwards mapped into correspondent created relations, as shown in Appendix E.

5.2.4 Using INGRES Data Manipulation Language

The data manipulation language (DML) supported by the INGRES DBMS is QUEry Language (QUEL). The QUEL DML is the language which the database programmer uses to cause data to be transferred between his or her program and the database. It is not a complete language by itself. It relies on a host programming language to provide a framework for it and to provide the procedural capabilities required to manipulate the data. Complete information on QUEL and INGRES appears in the INGRES reference manual (34). The use of QUEL DML is shown in both Appendix G and Section 5.3, respectively, illustrating the second and third levels of implementation (15).

5.2.5 Selecting Access Method Structures

In order to implement relations using the INGRES DBMS, three basic access method structures are available: heap, hash, and ISAM. The main characteristics of each one was taken in consideration, comparing the relative advantages and disadvantages of each option.

The heap access method structure is recommended for relations that use sequential retrievals for its data items. The REPORT relation from Appendix E is an example of a heap access method structure, assuming that each report must be retrieved in sequential order by report number, date, and time. INGRES always creates a relation assuming a heap access method structure. When a relation is implemented as heap, it can be changed to hash or ISAM.

The hash access method structure is recommended for relations that use random retrievals for its data items. It is advantageous for locating tuples or relation rows identified by an exact value. The REPORTID relation from Appendix E is an example of hash, assuming that each report number must be retrieved randomly to get its corresponding name.

The ISAM stands for indexed sequential access method structure and is useful for both sequential retrieval, when all data from the relation need to be retrieved, and for random retrieval when specific data from a relation need to be retrieved. Since the ISAM directory must be searched to locate tuples, it is never as efficient as hash. The MISTYPE relation from Appendix E is an example of ISAM assuming that both sequential and random access must be used for retrievals (16).

5.2.6 Main Menu and Report Generator Implementations

The projected layouts shown in Figures 18 Report #7 Consumed Items and Figure 20 System Main Menu were partially implemented (19) (34) as an example of a database report generator through the application program shown in Appendix F. Figure 21 and Figure 22 respectively show these implemented layouts.

```
BRAZILIAN AIR FORCE

FLYING UNIT OPERATIONAL CONTROL SYSTEM

  ***  MAIN MENU  ***

-------------------------------------------------------------------------------

        OPTIONS:
            1          Data Entry
            2          Report Generation
            3          Update Transaction
            4          Query Transaction
            5          Help
            6          Quit

-->  2
Please input report number and flying unit number.  7 2132732
Please input start and end dates (YYMMDD).  841130 850315
```

Figure 21 - The Partial Implemented System Main Menu

```
***********************************************************************************
BRAZILIAN AIR FORCE                                          R_DATE : 841107
FLYING UNIT OPERATIONAL CONTROL SYSTEM                       R_TIME : 1726
FLYING UNIT NO:     2132732                                  R_SDATE : 841130
REPORT NUMBER:            7                                  R_EDATE : 850315
REPORT NAME: Items per Mission

***********************************************************************************
S_MTCODE    A_TYPE         A_NO S_DEPLOC  S_ARRLOC    S_TIME S_ITCODE      S_ITCOQY

combat      mirage         2120  sbbr      sbbr         0.7  gunshot           23
grndsupp    mirage         2120  sbbr      sbsc         6.2  bombmK76           6
combat      mirage         2120  sbsc      sbsc         2.3  bombmk102          2
grndsupp    f-16           2220  sbbr      sbbr         2.1  gunshot           36
intercept   mirage         2121  sbbr      sbbr         2.7  miss77             2
intercept   mirage         2121  sbbr      sbrj         1.3  miss24             1
intercept   mirage         2121  sbbr      sbsc         2.3  rocket5            2

***********************************************************************************
Mission    combat      Total Sortie Time:       3.00
Mission    grndsupp    Total Sortie Time:       8.30
Mission    intercept   Total Sortie Time:       6.30

Total  gunshot      Consumed:         59
Total  bombmK76     Consumed:          6
Total  bombmk102    Consumed:          2
Total  miss77       Consumed:          2
Total  miss24       Consumed:          1
Total  rocket5      Consumed:          2
***********************************************************************************
```

Figure 22 - The Implemented Report Generator using Report #7

5.3 The Third Level of Implementation - Queries

This section deals first with a practical method to analyze, build and retrieve database query transactions. Then, following the created method, queries are developed using the relational database designed in Chapter IV, implemented in Chapter V, and shown in Appendix G. After that, more advanced query retrievals involving modified database relations are developed (15) (16) (34).

5.3.1 Building Query to Retrieve (BQtoR) Method

To retrieve queries from the database implemented in Chapter V, a method for Building Query to Retrieve, named "BQtoR" for short, was created. This method represents an attempt to standardize, as much as possible, an efficient and quick process to analyze, design and implement database queries. It should be used by the database designer.

"BQtoR" refers to the process of generating a query from the user request and consists of the following seven basic steps: a) understanding the user's query request, b) query specification, c) query analysis, d) query formulation, e) query implementation, f) query test, and g) query retrieval.

a) Understanding the user's query request refers to not only receiving a new query request, but also studying and understanding which are the actual query purposes in terms of data item retrieval needs.

b) Query specification identifies detailed data items required to support various decisions (who needs to make which type of decisions).

c) Query analysis identifies the process of determining how to get a specific data item using implemented or modified database relations.

d) Query formulation refers to the process of writing the specific query using a database data manipulation language (DML) and constructing a user friendly query to permit the easiest understanding and retrieving.

e) Query implementation refers to the process of loading the query into a specific DBMS. Depending upon its frequency a simple and standard application program can be used for each query implementation.

f) Testing the query refers to the process of checking and eliminating possible error conditions in the specific query data manipulation language or in the query application program.

g) Query Retrieval refers to the process of obtaining the final and logical answer for a formulated query, able to support a decision maker with specific information.

Two query transactions were initially selected to be implemented, applying the "BQtoR" method. These queries were requested based upon the author's previous experience in the flying unit operations environment.

5.3.1.1 The First Query Implementation

"Which AIRCRAFT TYPE have consumed the item MISS77 ?"

a) This query transaction was understood as a necessary user request to implement, because it could support decision makers with useful data items already existent in the database.

b) It was determined that the flying unit operations manager, as one of the decision makers in this case, should know which "type of aircraft" have consumed a specific "type of item" (weapons or supporting items), in order to optimize the flying unit's aircraft employment.

c) This query transaction involves attributes from two different relations: AIRCRAFT and MISTYPE. The attribute aircraft type (A_TYPE) belongs to AIRCRAFT relation and sortie item code (S_ITCODE) belongs to MISTYPE relation. Both relations contains a common attribute aircraft tail number (A_NO). In order to sucessfully retrieve the desired data items, these two relations must be joined.

d) The query transaction was formulated as follows using the query language QUEL, the INGRES data manipulation language (DML) (34).

```
* range of a is aircraft
* range of m is mistype
* retrieve (a.a_type) where
*             a.a_no = m.a_no and
*             m.s_itcode = "miss77"
*  g
```

e) The query transaction code was implemented using the INGRES DBMS, the relational database management system installed in the AFIT VAX 11/780.

f) The process was sucesfully tested using different data items and comparing with a manual processing.

g) Finally, the following answer was retrieved from the database, demonstrating the implementation of the first query transaction from the BAF Flying Unit Operational Control System.

```
   a_type
------------
|  mirage  |
------------
 (1 tuple)
```

5.3.1.2 <u>The Second Query Implementation</u>

```
------------------------------------------------
| "What CREW MEMBERS                           |
| are able to perform INTERCEPTION missions ?" |
------------------------------------------------
```

a) This query transaction was considered a necessary user request to implement, using existent database attributes and relations. It was understood that the terms "what CREW MEMBERS" actually refers to "what CREW MEMBER SSN (C_SSN) and LAST NAMES (C_LNAME).

b) It was determined that it is important for the flying unit operations manager to know what crew members are able to perform certain types of missions, in order to maintain their levels of proficiency in performing specific types of missions.

c) This query transaction involves attributes from three different relations: CREW, MISTYPE, and FLYUNIT. The attribute crew member last name (C_LNAME) belongs to the CREW relation and the attribute mission type code (S_MTCODE) belongs to MISTYPE relation. The first two relations CREW and MISTYPE do not have commum attributes. A third relation FLYUNIT contains both attributes C_LNAME and S_MTCODE. In order to sucessfully retrieve the requested data items, these three relations must be joined.

d) The query transaction was formulated as follows using INGRES QUEL.

```
* range of c is crew
* range of f is flyunit
* range of m is mistype
* retrieve (c.c_ssn, c.c_lname) where
*          c.c_ssn = f.c_ssn and
*          f.a_no = m.a_no and
*          m.s_mtcode = "intercept"
*  g
```

e) The query transaction code was also implemented using the INGRES DBMS.

f) The process was sucessfully tested with different C_LNAME and S_MTCODE and compared with manual processings.

g) The following answer was sucessfully obtained from the query transaction.

c_ssn	c_lname
391801970	john
591811724	david

(2 tuples)

5.3.2 Implementing Advanced Queries

After analyzing the partially implemented Flying Unit Operational Control Database System, some queries involving modified relations and nonexistent database attributes were developed. These types of query transactions could not be retrieved using the traditional approach of the current file management system.

In order to take advantage of the efficient relational database structure, three advanced query transactions were selected to be implemented in this section. They were considered advanced query transactions, because of their nature of being generated from modifications to the database relational structure. The advanced query transactions were also generated following the BQtoR method.

5.3.2.1 The First Advanced Query Implementation

"Which AIRCRAFT from which BAF FLYING UNITS are available to perform FORMATION missions on January 14, 1985 at 10 o'clock ?"

a) This query transaction was considered a necessary user request to implement. It was understood that the terms "Which AIRCRAFT" in this query refer to which aircraft tail numbers (A_NO), and also that the date and time specified are to provide the decision maker with the important information about all aircraft status from a flying unit at a certain point in time.

b) It should be very usefull for the BAF operational control manager, as an important decision maker, to know what crew members from which BAF flying units are qualified to perform which mission code type at any time. It was determined that, if sucessfully implemented this specific type of query retrieval could not only rationalize and optmize the flying unit resource employments, but could also provide a better operational control over the BAF Flying unit's activities.

c) This query transaction involves three relations. Two of them, FLYUNIT and MISTYPE, already exist in the database. A third and new relation must be created and included in the summary of 3NF (Section 4.2.5) and database conceptual model (Section 4.2.6). This new relation received the number 310, meaning the tenth database relation composed of three key data items. It was named ACFTAVAL (aircraft availability) as shown in Appendix D and F. The new ACFTAVAL relation is composed of the following attributes: aircraft tail number (A_NO), aircraft available date (A_AVDATE), aircraft available time (A_AVTIME), aircraft available period (A_AVPERD), and aircraft available status or situation (A_AVSIT). The relationship between attributes A_AVPERD and A_AVSIT; and A_NO, A_AVDATE, and A_AVTIME is one-to-many because several aircraft numbers with different available dates and times may be available in the same period with the same situation. The ACFTAVAL relation is shown in the Appendix F and should be represented as follows in Table 7.

310) A-NO,A-AVDATE,A-AVTIME <--<---> A-AVPERD,A-AVSIT

An Acftaval Table Sample

A-NO	A-avdate	A-avtime	A-avperd	A-avsitu
2120	050114	100000	2	available
2121	050107	70000	1	situationc
2122	050114	100000	2	available
2220	050103	63000	1	available
2223	050112	210000	3	situationa

Table 7 - The Created Acftaval Table

d) The query transaction was formulated as follows using INGRES QUEL.

```
* range of v is acftaval
* range of f is flyunit
* range of t is mistype
* retrieve (f.f_unitno, f.a_no) where
*           f.a_no = v.a_no and
*           v.a_avdate = 850114 and
*           v.a_avtime = 100000 and
*           v.a_no = t.a_no and
*           t.s_mtcode = "formation"
*  g
```

e) This query transaction code was also implemented using the INGRES DBMS.

f) The process was sucessfully tested using different A_NO, A_AVDATE, A_AVTIME, A_AVPERD, and A_AVSIT, and sucessivelly compared with manual processings.

g) The following answer was sucessfully obtained from the query transaction.

f_unitno	f.a_no
03110234	2223

(1 tuple)

5.3.2.2 The Second Advanced Query Implementation

```
"What qualified CREW MEMBERS,
from which BAF FLYING UNITS,
are able to perform NOW,
a COMBAT mission ?"
```

a) This query transaction was also considered a necessary user request to implement. This second advanced query transaction introduces the term "NOW" that was considered very important and useful for the advanced query retrievals developed in a flying unit operational control system environment. It involves two inferential groups of terms that need to be properly understood before being implemented. In the first group of terms "What qualified CREW MEMBERS", the specific term "qualified" actually refers to those "crew members pursuing the functional qualification code (C_FQCODE) of 1P meaning first pilot, IN meaning instructor pilot, 1F meaning first flight engineer, and IF meaning instructor flight engineer. In the second group of terms "are able to perform NOW", the specific term "NOW" actually refers to the present date and time. The purpose of this query transaction is unquestionably important, because its sucessful retrieval should support immediate actions in the decision making process.

b) It should be very important for the BAF operational control manager to know what qualified crew mwmber from which BAF flying units are available to perform a specific type of mission at a certain point in time. In fact, this specific query transaction, sucessfully retrieved, could represent an employment optimization of qualified resources in the system environment.

c) This query transaction involves the following three relations: CREW, MISTYPE, and FLYUNIT. Neither new nor modified relations should be used. Just the existent ones. But due to its inferential nature, similarity with more advanced queries, importance and frequency of retrieval, it was also considered an advanced query.

d) The query transaction was formulated as follows using INGRES QUEL.

```
* range of c is crew
* range of f is flyunit
* range of m is mistype
* retrieve (c.c_ssn, c.c_lname) where
*           c.c_fqcode = "1p" or
*           c.c_fqcode = "in" or
*           c.c_fqcode = "1f" or
*           c.c_fqcode = "if" and
*           c.c_ssn = f.c_ssn and
*           f.a_no = m.a_no and
*           m.s_mtcode = "combat"
*  g
```

e) This query transaction code was also implemented using the INGRES DBMS.

f) The implementation was sucessfully tested using different C_CSS, C_LNAME, C_FQCODE, A_NO, and S_MTCODE.

g) The following answer was sucessfully obtained from the query transaction.

c_ssn	c_lname
591811724	david

(1 tuple)

5.3.2.3 <u>The Third Advanced Query Implementation</u>

> "What qualified CREW MEMBERS and which AIRCRAFT,
> from which FLYING UNITS,
> can be employed NOW (i.e., at 0630 Jan 03, 1985),
> to perform FORMATION mission in REGION d ?"

a) This query transaction was also considered a necessary user request to implement that involves the ACFTAVAL relation recently created in Section 6.2.1 and also a modified relation. It was understood that the terms "CREW MEMBERS" refers to the crew members SSN and last name, the term "AIRCRAFT" refers to which aircraft tail numbers, and the terms "in REGION d" refers to the geographical region where flying units are assigned to operate. This query intends to provide the decision maker with important information about flying unit employment in different regions.

b) It should be very useful to the BAF operational control manager to know what flying unit crew members and aircraft can be employed to perform specific types of missions in different regions at different time. It was determined that, if sucessfully implemented, this query transaction could substantially improve the efficiency of the decision making process involving the operational control of the BAF flying units activities.

c) This query transaction involves the following four relations: CREW, MISTYPE, ACFTAVAL, and FLUNIT. The relation number 102 from Section 4.2.5, Summary of 3NF relations represents the second created database relation with one key data item. For this query retrieval, it should be modified by appending a new attribute flying unit region (F_UNITRG). In order to append the F_UNITRG to the database, the Brazilian air space was divided into different regions according to which flying units were assigned to operate, as shown in Figure 23. The flying unit region (F_UNITRG) was considered a non-key attribute. The relationship between the key data item F_UNITNO and the non-key data items C_SSN, A_NO, and F_UNITRG is one-to-many, because a flying unit may have several crew members flying different aircraft, performing different missions in different regions. The modified FLYUNIT relation was renamed FLUNIT. Its implementation is shown in Table 8 and Appendix F. Its 3NF representation shown below should substitute for relation 102 from Section 4.2.5.

102) F-UNITNO <--<-----> C_SSN, A_NO, F_UNITRG

A Flunit Table Sample

F-UNITNO	C-SSN	A-NO	F-UNITRG
02132732	291801970	2120	a
02132732	291801970	2220	a
02132732	391801970	2121	b
02132733	491801923	2122	c
03110234	591811724	2223	e
03110234	691801613	2301	d

Table 8 - The Modified Flunit Table

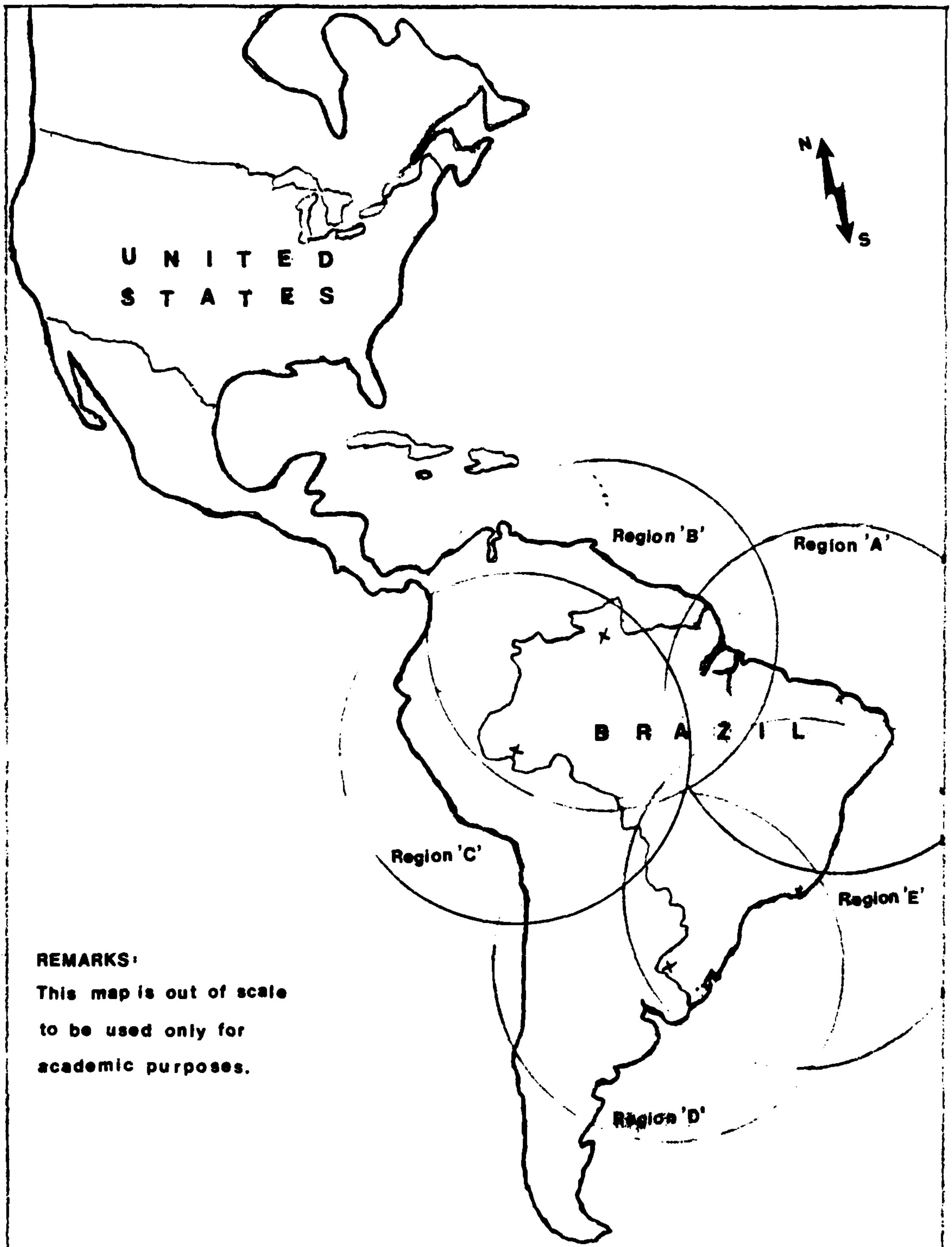


Figure 23 - The Brazilian Regions for Query Retrievals

d) The query transaction was formulated as follows using the INGRES QUEL.

```
* range of c is crew
* range of f is flunit
* range of m is mistype
* range of v is acftaval
* retrieve (f.f_unitno, c.c_ssn,
*           c.c_lname, v.a_no) where
*           c.c_fqcode = "lp" or
*           c.c_fqcode = "in" or
*           c.c_fqcode = "lf" or
*           c.c_fqcode = "if" and
*           c.c_ssn = f.c_ssn and
*           f.f_unitrg = "d" and
*           f.a_no = v.a_no and
*           v.a_avdate = 850103 and
*           v.a_avtime = 063000 and
*           v.a_avsitu = "available" and
*           v.a_no = m.a_no and
*           m.s_mtcode = "formation"
*  g
```

e) This query transaction code was implemented using the INGRES DBMS.

f) The implementation was sucessfully tested using different F_UNITNO, C_SSN, C_LNAME, A_NO, C_FQCODE, F_UNITRG, A_AVDATE, A_AVTIME, A_AVSIT, and S_MTCODE.

g) The following answer was sucessfully obtained from this query transaction.

f_unitno	c_snn	c_lname	a_no
03110234	691801613	Foster	2301

(1 tuple)

The implementation of these advanced query transactions in this thesis constitute only a sample of what could be done, in order to improve the efficiency levels of the flying unit operational control system environment.

VI. Optimizing Database Query Retrievals

In this chapter an investigation of database query retrievals is performed considering the Artificial Intelligence (AI) approach and supporting concepts. Problems involving query retrievals are identified. Present trends and future directions for optimizing database "intelligent" retrievals are discussed. And finally, some ideas based on the LADDER-like system (17) (28) are presented. Examples of database query retrievals are projected for a hypothetical BAF FLUNITOC Front-end system using natural language.

Although the Brazilian Air Force FLying UNIT Operational Control (BAF FLUNITOC) database system represents an improvement compared with the previous file management system, there are still some problems to overcome. For instance, consider the third advanced query transaction implemented in Chapter V. Suppose instead of NOW (representing the present time and date), the decision maker desires the same query projected for eight hours ahead (Considering in this case the projected window of aircraft availability as 8 hours). The system will not be able to handle the new situation except by processing the whole query again. This is considered a waste of time, or more precisely, a system limitation.

Nowadays, more user-friendly and human-type systems have been required for dynamic systems similar to the BAF FLUNITOC. Recently, systems with some degree of reasoning and deductible capability, able to process English-like natural languages have been considered as optimum solutions. They are frequently recognized necessary and important to be adopted for future applications. For these types of systems, a user used to be required to explicit statements to specify a formal query. It would be better, however, if the user could write or even talk with the computer using some English-like natural language (2).

6.1 The AI Approach for Query Retrievals

A system that uses a natural language is an information-processing activity of great complexity. Endowing computers with this ability has long been a major goal of research in Artificial Intelligence (AI), also called machine intelligence, a branch of experimental computer science that studies the nature of knowledge and its manipulation. There are many applications of AI available today including natural language processing, intelligent retrieval from databases, expert consulting systems, theorem proving, robotics, automatic programming, and combinatorial and scheduling problems (22).

For this thesis, only some of the AI aspects are considered, such as the design of intelligent computer systems, its associated characteristics with intelligence in human behavior to understand language, and database query retrievals to solve decision making problems.

This Chapter presents an investigation of the AI approach, theory and concepts, in order to project future growth of the BAF FLUNITOC system in its right direction.

First some major foundations and concepts of the AI theory are discussed to support future database query retrievals. Then the AI approach is roughly compared with the DSS approach presented in Chapter III. After that, several implemented systems which have used the AI approach and techniques are mentioned. Some existing ideas considered of wide value in both military and non-military applications are explored. Finally, the main goal, present trends and future directions for a Front-end BAF FLUNITOC dialog system are discussed.

6.1.1 AI General Concepts

In order to understand database query retrievals using the AI approach, some concepts from AI theory such as natural language, computational linguistics, symbolic language, and other related ideas are discussed in this section.

a) Natural Language (NL)

Natural Language (NL) processing systems are more desirable than other possible database front-end systems, such as touch-panel menu-selection schemes or systems using special purpose data languages. A database front-end processing systems refers to any processing of data before the main work of a DBMS begins. An edit program that checks input data for acceptable ranges of values could be considered a simple example of front-end processing.

A system that can phrase and verify complicated questions relatively easily with a natural language have been considered the most important type of database front end. Current touch-panel menu-selection systems seem best at rapidly selecting specific files to display, either in their entirety or subject to simple restrictions. But, if one wishes to display only file elements which simultaneously satisfy several predicates or relations which require logical combinations of several files, more expensive programming is required than can be done simply by menu selection (28:526,539).

Special purpose data languages can deal with complex searches and combinations of relations, but learning such languages requires extensive training, and even then, input programs may be difficult to formulate, verify, and troubleshoot.

The advantage of NL systems in phrasing questions will become even greater when the ability to handle vague and complex questions is more fully developed (31).

Research centers like MIT, Carnegie Mellon, Stanford and other U.S. Universities have been successfully building a foundation of AI theory, techniques and tools since the late 1950s. The commercial availability of hardware and software tools make it possible to develop economically justifiable AI applications for a wide range of end user organizations previously served only by more traditional data processing methods (17).

b) Computational Linguistics

Computational Linguistics is a science at the juncture of AI, Philosophy, Linguistics, and Psychology that has the central objective of understanding the computational mechanism that underlie the use of natural languages. The two primary goals of Computational Linguistics are: to understand how humans communicate and to create machines with human-like communication skills.

The first is a scientific goal of understanding people by having better notions of the use of NL and the mental process involved, by better communicating in language skills, and even designing more efficient intercomputer communications. The second goal is an engineering one pursued for practical purposes to create machines that can communicate with people in languages they already know. Today, only computer programmers, representing a small segment of the population, can communicate with computers. Progress in computational linguistics is facilitated by pursuing both of the above goals simultaneously.

The advent of machines that understand NL will make it possible for anyone to directly use powerful computational systems. The ultimate goal of creating machines that can easily interact with people remains far off, awaiting both improved information processing algorithms and alternative computing architectures. However, progress in the last decade has demonstrated the feasibility of employing today's computers to deal with NL.

Furthermore, microcomputer implementation of these limited language-processing techniques is leading to more practical and cost-effective systems. An important application of NL processing is as a part of large computer-based systems. Providing answers to questions by accessing large databases represents just one practical application of computational linguistics.

c) <u>Symbolic Languages</u>

Prolog is a logical programming language developed by a group at Marseille, France, headed by Alain Colmerauer. It has been the pivotal software tool for AI in Europe and Japan. As a symbolic language, Prolog is based on ideas proposed by Robert Kowalski at the University of Edinburg, who suggested using logical inferences as a form of computation. Prolog uses familiar questions and answers in interaction with a common database. A user poses a question to the computer using Prolog and its database responds. Programs in Prolog are rules stored in the database. The rules transform input to output, thus alleviating the need to explicitly state how to use knowledge in the database to answer a question (30).

The List Processing Programming Language (LISP) is a symbolic language that has been the pivotal software tool for AI in the USA. The original version of LISP was developed by John McCarthy in 1957. Unlike the most familiar traditional programming languages such as BASIC, FORTRAN, COBOL, Pascal, etc., LISP deals with complex objects, not just numbers. Therefore it lends itself to the development of flexible systems that can accommodate ambiquities, infer relationships between data, and even perhaps learn.

The "LISP-machine" or symbolic processor is a computer system whose logical architecture is specifically designed to support economical AI program development. AI is expanding rapidly and computer industry analysts predict it may account for 50% of all EDP by the end of the 1990s. The Japanese have been aggressive with their AI research program in the Fifth Generation Computer project. Electronic circuits photographed, reduced and etched into silicon wafers have made the computer smaller, more powerful, and cheaper. AI has eaten up a lot of memory, but memory is cheap today (18).

Symbolic processing capability is the idea behind AI techniques that makes people more productive. Computers understand only two states, on and off, the binary coding system of zero and one. Conventional or formal computer programming assigns numerical equivalents to defined pieces of data and then assigns those numerical equivalents to defined files. Data definitions and the allowable relationships between data items and files are narrow, literal and unforgiving.

People think in terms of complex symbols, and symbolic logic is tremendously economical. People learn more than they can consciously remember. In a recent seminar offered through Worchester Polytechnic Institute, from the Office of Advanced Systems and Software Technologies, Gould, Inc., Richard Morley and William Taylor summarize, "Computers have infinite memory and limited processing power. People have infinite processing power and limited memory (18)."

Symbols and their properties are maintained in a relational database in which the interrelationships possible among them are expanded under the operation of general rules rather than limited by predetermined data structures. Symbolic processing has two basic advantages over numerical processing: flexibility and the ability to accomodate uncertainty and extreme complexity. Symbolic processors measure their speed in Logical Inferences Per Seconds (LIPS) rather than in arithmetic operations per second (18).

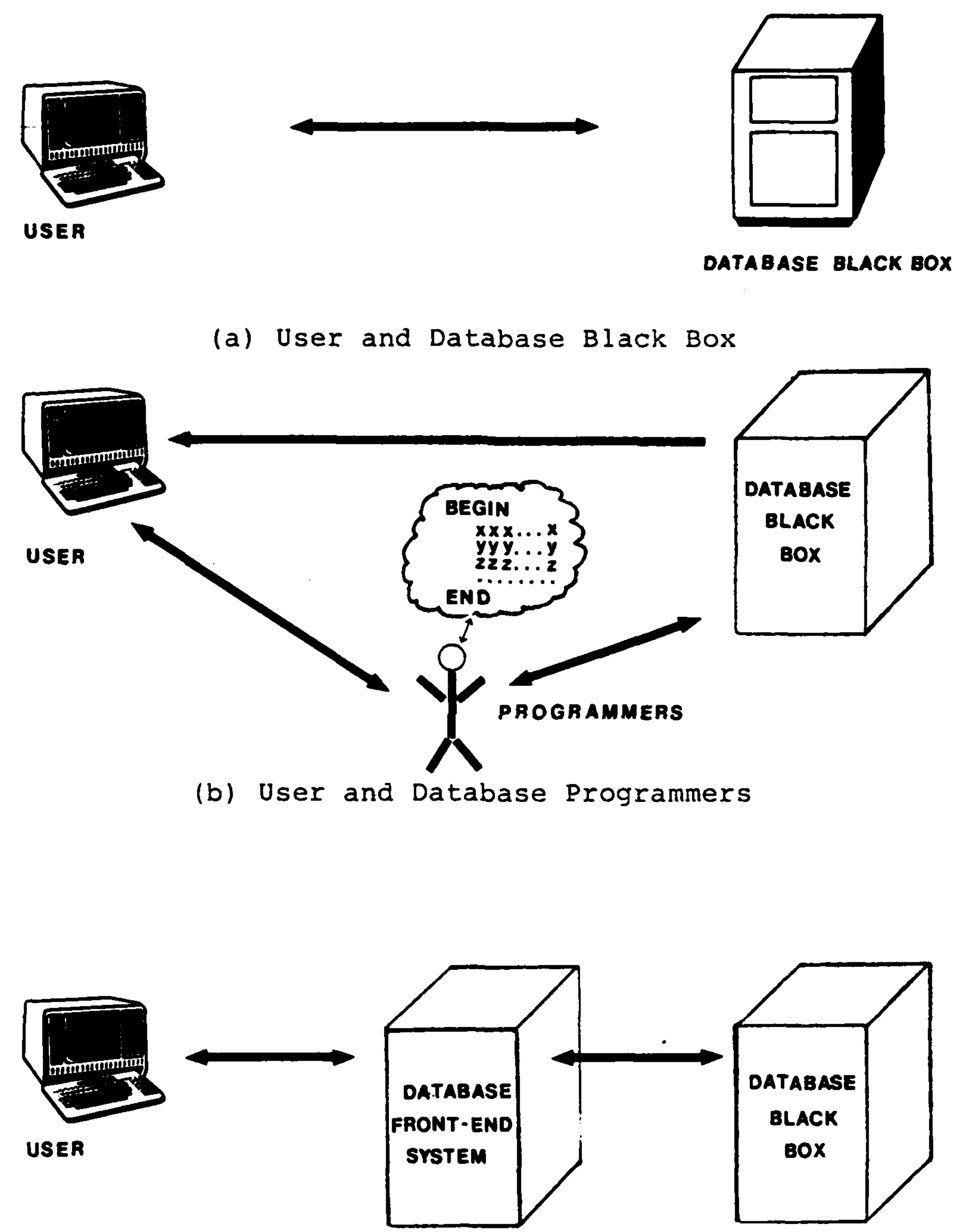
In order to support database query retrievals using the AI approach many other important AI concepts indirectly applicable in this thesis work have been explored. Some of them can be found in references 17, 29, 30, 32, and 33.

6.2 Present Trends for Query Retrievals

There are several problems that confront the designer of intelligent database query retrieval systems. One is the tremendous problem of building a system that can understand queries stated in an English-like NL. If the problem of language understanding is overcome by specifying some formal, machine understandable query language, the problem remains how to deduce answers from stored facts. Understanding a query and deducing an answer may require knowledge beyond that explicitly represented in the subject domain database, because commom knowledge that is typically ommitted in the subject domain database is often required.

For example, consider the third advanced query transaction implemented in Chapter V, an intelligent system ought to be able to deduce the answer "F-15" to the query "Which aircraft type does Foster fly ?" Such a sytem would have to know somehow that the term "fly" indicates a relationship between a crew member and an aircraft type.

One of the system design problems that invites the methods of AI is how common knowledge should be represented and used. Also one of the most important and feasible areas for the application of NL processing is acessing data in databases. For several years, a database has been seen as a black box as shown in Figure 24(a), from where users had to



(a) User and Database Black Box

(b) User and Database Programmers

(c) User and Front End Database System

Figure 24 - Users, Programmers, and Database Relationships

extract important information.

Nowadays, to produce timely answers to questions and quickly clear up problems as to how a decision maker's question is to be interpreted, users obtain information from the computer through programmers as shown in Figure 24(b). Programmers are considered interpreters who translate queries into a form the machine understands. Research groups around the world are attempting to automate programmers' tasks to make the interpretation process simpler.

Researchers understand that systems should be as human independent as possible. Recently, front-end database systems have been designed to overcome this problem as shown in Figure 24(c).

In order to accept simple English queries specifying what information a user wants, systems already exist capable of generating fairly complex programs specifying how computers are to retrieve information. Two problems are confronted together: 1) the system must translate from English or any other language (e.g. Portuguese0 into a formal language, and then, 2) convert a statement of what is wanted into a statment of how to get it.

Some examples of such a systems are: a) SHRDLU, b) LSNLIS, c) PLANES, and d) LADDER (28).

a) A system called SHRDLU completed in 1971 by Terry Winograd at MIT AI Laboratory is able to answer quite complex questions about a simple database representing a visual scene in a block world. It can carry out a dialogue with the user disambiguates sentences and handles pronouns and phrase references (28).

b) The Lunar Sciences Natural Language Information System (LSNLIS) of William Woods et al. answers questions about a fairly large database of samples of lunar rocks and soils. The Lunar system is able to accept grammatically complex sentences involving nested dependent clauses, comparative and superlative adjective forms, and some types of anaphoric references. This system can answer questions like: "What is the average concentration of aluminium in high alkali rocks ?", "Give me all model analysis of lunar fines.", and "List all rocks which contains certain substances". It uses an Augmented Transition Network (ATN) to parse sentences and then generates a formal query by patching together the code fragments which represent each phrase in the sentence (28).

c) The Programmed LANguage-based Enquiry System (PLANES) developed under Office of Naval Research sponsorship, like Lunar, was designed to operate on a real database. The PLANES database query language and general organization are fairly direct implementations of Codd's ideas developed by Palermo (28). In order to operate, PLANES uses a database from the NAVY 3-M Database for Aircraft Maintenance, a Statiscs base to "optimize" the order of search, and Flight Information extended over a period of time. The system can handle pronouns and referential phrases, common sense questions, search strategies, patterns of data recognition, and has many other features (28).

d) The LADDER system developed at SRI International is a NL processing system to access large databases capable of translating a question such as: "To what country does the fastest submarine belong ?" (28).

The main point is that LADDER-like systems exist today that are capable of accepting simple English queries specifying "what" information a user wants and then generating fairly complex programs specifying "how" the computer is to retrieve the information.

A LADDER-like system could be designed in the near future, as a front-end database system for the BAF FLUNITOC database system. In order to explore this idea, and reach a better understanding of the type of NL processing possible using the AI approach and technology available today, a sample of LADDER-like dialog for a hypothetical BAF Front-end FLUNITOC is presented in Figure 25. In this sample, statements in capital letters mean a user dialog and statements starting with "%" represent the system dialog (17).

Query 1, "Give me the length of the Mirage 2120", contains a misspelled word. A LADDER-like system corrects this automatically by typing a note to the user directly below the input line. Once the spelling error is corrected, an analysis of the input is completed and a LADDER-like system types the message "Parsed !" displaying its interpretation of the user's query. The system then would call the remote FLUNITOC database system to retrieve the correct answer, "Length = 50", meaning that the length of the Mirage 2120 is 50 feet.

```
1 -  GIVE ME THE LENGTH OF THE MIRRAGE 2120.
%                   Spelling --> Mirage
% Parsed !
% Give the length of the aircraft type (A_TYPE) Mirage
% which tail number (A_NO) is 2120.
% Length = 50

2 -  WIDTH AND DRAFT
% Parsed !
% Trying Ellipsis !
% Give the beam and draft of the aircraft Mirage 2120
% Beam = 60
% Draft = 26

3 -  WHO ARE ITS CREW MEMBERS ?
% Give the SSN (C_SSN), rank (C_RANK), and last name
(C_LNAME)
% of the crew members flying NOW the Mirage 2120.
% First Pilot = 291801970 Major Cunha
% Second Pilot = 391801970 Capt John

4 -  WHAT ABOUT 2223 ?
% Parsed !
% Trying Ellipsis on 2223.
% Who are the crew members of the aircraft F-16 2223 ?
% Give the SSN (C_SSN), rank (C_RANK), and last name
% (C_LNAME)
% of the crew members flying Now the F-16 2223.
% First Pilot = 59181124 Capt David

5 -  WHAT ITEMS DOES IT CARRY NOW ?
% Parsed !
% Give the sortie item code (S_ITCODE) carried by F-16 2223.
% (S_ITCODE) = 2 Rocket1 and 2 Miss77
```

Figure 25 A Hypothetical BAF FLUNITOC Front-end NL Dialog

Query 2 is not a complete sentence. In fact, it makes no sense when considered in isolation. In the context of the preceding query, it is clear that the intent of the query 2 is, "Give me the width and draft of the aircraft Mirage 2120". Leaving out pieces of a sentence is called "ellipsis". Processing such elliptical inputs is more difficult that it may seem. The system has to avoid such interpretations as, "Give me the length of the width and draft", or even, "Give width and draft the length of the Mirage".

Query 3 illustrates the use of a pronoun, another linguistic construct that cannot be interpreted in isolation and query 4 demonstrates a more sophisticated form of ellipsis in which the analysis involves more than just grammatical substitution of the new input into the old one. This is accomplished in a LADDER-like system by looking for "What about" sentences at the beginning of an input, and then applying the standard elliptical technique to what follows. Finally, query 5 causes a body of information to be retrieved directly from the FLUNITOC database.

The computer capability shown in Figure 25 is of considerable practical utility. The original LADDER system deals with a relatively large and complex database including over 100 fields in fourteen files and has records for 40,000 ships. There are also several systems around the word that are capable of similar performances (17).

6.3 Future Directions for Query Retrievals

Considering the more advanced type of processing available in the LADDER-like systems, future directions for intelligent query retrievals for a hypothetical BAF Front-end FLUNITOC System can be easily visualized through the sample dialog presented in Figure 26.

Statements in capital letters mean a user dialog and statements starting with "%" represent the system dialog. Statements in square brackets replace lengthy material not relevant for the purpose of the dialog (17).

```
1 -  LET "SHOW THE FORCESTATUS OF THE MIRAGE 2120" BE LIKE
     "DIPLAY THE EMPLOYMENT AND READINESS CONDITIONS OF THE
     MIRAGE 2120.  PRINT ITS DESTINATION.  LIST AIRCRAFT IN
     ITS FLIGHT."
% Parsed !
% [ New production added to the system ]

2 -  SHOW THE FORCESTATUS OF F-16 2123
% Parsed !
% [ Question defined in query 1 for Mirage 2120 are answered
     for F-16 2123 ]

3 -  DEFINE "QUAIS SAO AS CONDICOES DE EMPREGO OPERACIONAL
     DO MIRAGE 2120 ?" LIKE "WHAT IS THE FORCESTATUS OF
     MIRAGE 2120?"
% Parsed !
% [ Production added to system ]

4 -  QUAIS SAO AS CONDICOES DE EMPREGO OPERACIONAL DO MIRAGE
     2120 ?
% Parsed !
% (S_DEPLOC) = sbbr
% (S_TIME) = 1000
% Endurance = 0200
% (S_ITCODE) = 2 Miss77, 6 Rocket2, and 90 Gunshot.
```

Figure 26 The BAF FLUNITOC Front-end Using Portuguese NL

This dialog was also adapted from a LADDER-like system to project a future BAF Front-end FLUNITOC using Portuguese as the NL.

In query 1 the user tells the LADDER-like system to make a certain input sequence equivalent to not just one but a whole series of questions. It is as if the user were writing small programs in English NL, using pronouns for formal parameters.

In query 2 a new construction is used, but with a different aircraft type and number than the one used to define the construction.

In query 3 the user tells the LADDER-like system a Portuguese paraphrase of the English question. This is an extreme example of this ability to accommodate user defined constructions.

In query 4 the question about Mirage 2120 is posed entirely in Portuguese.

The language processing capabilities demonstrated in Figure 26 seem quite adequate for a wide range of practical applications. But these capabilities are still far from a fluent use of natural language.

Fluent use of natural language by machines remains an elusive aspiration. In order to extract from databases timely answers to questions and clear up problems as to how a decision-maker's question is to be interpreted, the turnaround time must be cut from hours or days to seconds.

The development of a BAF Front-end FLUNITOC system, using the AI approach and Portuguese as a NL, seems the most likely direction to be followed for future intelligent query system retrievals. The main goal for the development of such a system should be to allow nonprogrammers to obtain information from a large database with a minimum amount of prior training or experience. This system should be able to understand to a substantial degree the Portuguese NL and should be able to help, guide, and train the user to frame requests in a form that the system can understand (24).

Although the computational cost for NL processing is relatively high when compared with machine languages, the introduction of Very Large Scale Integration (VLSI) technology promises to ease the attendant cost.

Processes previously performed only in the laboratory on research computers costing over one million dollars are becoming practical to be implemented on personal computers.

Considering future trends and directions, the expenses of programming will continue rising while computer hardware will continue droping in price. For some applications, it will be cheaper create systems using subsets of a NL than to train people to use formal languages.

The fluent use of NL by machines remains a long-term goal. To deal with significant fragments of language in specialized application areas, a number of practical mechanisms have been developed. The ability to communicate within such fragments is both sufficient for the task at hand and clearly preferable to forcing users to learn machine-oriented languages (17).

The author of this thesis believes that in comming years NL processing will be employed in an increasing number of practical applications enabling more and more users to interact directly and effectively with computer systems. Consequently, the Brazilian Air Force Flying Unit Operational Control System should not be ommited from this reality.

VII. Conclusion

In this chapter, the main research findings related to the design of the relational database for the Brazilian Air Force FLying UNIT Operational Control (BAF FLUNITOC) system are summarized. Recommendations for further research and following courses of action are indicated. Finally, the results obtained are briefly discussed.

7.1 Conclusions

The main research findings are summarized as follows:

1) The Decision Support System (DSS) approach used to analyze the overall system environment was fundamental in development of the BAF DSS and databases top-down plans.

2) The design of the Dialog Generator Management Software (DGMSW) Prototype was an important DSS tool to determine the magnitude of the problem when performing a macro-environment top-down analysis.

3) The functional analysis successfully performed was considered an essential factor to determine not only the potential databases to be developed, but also to find out which one should be implemented first.

4) The use of the relational approach in both the conceptual and logical model was considered a tactical design technique which tremendously facilitated the

database design.

5) The partial system implementation performed at three different system levels demonstrated the feasibility of the BAF systems.

   a) The first level of partial system implementation demonstrated the BAF DSS feasibility.

   b) The second level demonstrated the BAF FLUNITOC database system feasibility as a significant part of the first level of implementation.

   c) The third level demonstrated the database query retrieval feasibility as the most important part of the second level of system implementation.

6) The Artificial Intelligence (AI) approach for database query retrievals projected for the hypothetical BAF FLUNITOC Front-end system demonstrated how the author of this thesis sees present trends and future directions for optimizing database query retrievals.

7) Finally, a major advantage gained from the design and partial implementation of this application was the possibility that it could be considered as a relational database generator for the BAF. In other words, the system demonstrates the feasibility of this concept as a reference for future relational database designs for the BAF.

7.2 Recommendations

As a natural consequence of the research developments, some suggestions are stated in the form of recommendations.

1) Research should continue beyond this thesis, in order to project this database design from the Flying Unit Operational Control to the Air Force Operational Control System level.

2) The design replication of the BAF FLUNITOC system running in a microcomputer environment, supported by improved DBMS under integrated and distributed databases, should bring future perspectives for this system.

3) The design of the DGMSW as the main strategy for DSS implementation should be continued.

4) The DSS and the database top-down plans should be immediately adopted in the BAF in order to control, discipline, and organize database growths.

5) Research in database query retrieval optimization should continue beyond this thesis investigation, by developing a Front-end system for the BAF FLUNITOC system.

6) The BAF should adopt a relational database management system (DBMS) in order to support the implementation of the FLUNITOC system, the development of potential application databases, and other research.

7) The current BAF Flying Unit File Management System

should be converted to the BAF FLUNITOC and other application database systems, following this thesis work.

The author recommends that a high level of importance be given to the following items, which, although not covered in this thesis, are directly related to the future implementation of the FLUNITOC system in the BAF. Database security restrictions should be enforced to the system. A computer network should be designed in order to support the system spread around the distant Brazilian regions, this network should be designed supporting future database developments according to the BAF database top-down plan and supporting integrated and distributed databases. Finally, the research and the development of computer systems directly applicable in the BAF environment is strongly recommended for the Brazilian officers taking the AFIT GCS program.

7.3 Results and Discussion

In order to preserve national sovereignty and guarantee internal and external country security; in addition to modern aircraft, high level of technology, qualified and trained personnel for operational employment, an Air Force should have efficient systems to control its flying units operations. Otherwise, it will not be as effective.

In an attempt to develop one of these efficient systems, this research sucessfully addressed the following main issues:

1) The BAF system requirements specifications;

2) The BAF DSS and database systems top-down plans;

3) The functional analysis of a system environment;

4) A database design approach and methodology;

5) The partial system implementation in three different levels;

6) The "Build Query to Retrieve" (BQtoR) creared method; and finally,

7) A hypothetical BAF FLUNITOC Front-end system using Portuguese as natural language (NL).

As an overall picture, the final results of this thesis research represent a pioneering effort to design the first relational database system to be implemented in the Brazilian Air Force.

Appendix A

The BAF File Management System Reports

The objective of this Appendix is to present the current BAF file management system report descriptions and layouts. These reports are considered batch processing because at least one application program is needed to generate one report at a time through the computer. They differ from on-line processing reports. In on-line processing, many reports can be generated at the same time by directly accessing a central processing site concurrently from several users terminals at separate locations.

Each report depics a user view and was considered as a starting point for the design process of the BAF FLUNITOC database system (9).

The following fourteen report layouts and descriptions represent all information needs of the current file management system and were studied and analyzed as system requirements:

Report #1 - Individual Flight Record,

Report #2 - Mission Type Summary,

Report #3 - Crew Member's Summary per Aircraft Type,

Report #4 - Missions Summary per Aircraft number,

Report #5 - Missions Summary per Administrative Unit,

Report #6 - Missions Orders Numbers,

Report #7 - Consumed Items per Mission,

Report #8 - Aircraft Numbers Summary Totals,

Report #9 - Aircraft Numbers Status,

Report #10 - Consumed Item per Aircraft,

Report #11 - Consumed Items Quantity,

Report #12 - Crew Member's Summary,

Report #13 - Crew Member Individual Totals, and

Report #14 - Crew Member's Totals Summary.

Report #1 - Individual Flight Record

This report consists of mission sorties performed by a specific crew member. Every flying unit may have several different crew members at a certain point in time, and this report is printed one for each crew member on a monthly or periodic basis. This report layout is shown in Figure 27.

```
****************************************************************
*BRAZILIAN AIR FORCE                        R_DATE : 02/01/85*
*FLYING UNIT OPERATIONAL CONTROL SYS        (Report Date)    *
*F_UNITNO: 2732-1353                        R_TIME : 00:00:00*
*(Flying Unit Number)                       (Report Time)    *
*R_NO : 01                                  R_SDATE: 01/01/85*
*(Report Number)                            (Rep Start Date) *
*R_NAME: INDIVIDUAL FLIGHT RECORD           R_EDATE: 01/31/85*
*(Report Name)                              (Report End Date)*
*                                                            *
*(Crew Member Social Sec. No.) C_SSN   : 291-80-1970         *
*(Crew Rank, Speciality) C_RANK, C_SPEC: Major, Pilot        *
*(Crew Last Name) C_LNAME              : Cunha               *
*(Crew Complement Name) C_CNAME        : Adilson Marques da*
*------------------------------------------------------------*
*   S_DEPDAT     S_CODENO    S_MTCODE    A_TYPE    A_NO      *
*   (Sortie      (Sortie     (Sortie     (Acft     (Acft     *
*   departure     Code       Mission     Type)     Number)   *
*   date )        number)    Type Code)                      *
*                                                            *
*   01/01/85     10408701     07FM03     F-103B     2120     *
*   01/01/85     10408702     14TG00     F-103B     2120     *
*   01/02/85     10408901     07FM03     F-103E     2123     *
*   --/--/--     --------     ------     ------     ----     *
*     ...          ....         ..        ....       ..      *
*------------------------------------------------------------*
*S_XFUNCT  S_DEPLOC  S_ARRLOC  S_TIME  S_LANDNO  S_ICOND    *
*(Sortie   (Sortie   (Sortie   (sortie (Sortie   (Sortie    *
*executed  departure arrival    time)  landing   instrum.   *
*function) location) location)         number)   condit.)   *
*                                                            *
*   1P       SBRJ      SBBR      1.5      02         R      *
*   IN       SBBR      SBRJ      1.6      03         R      *
*   IN       SBRJ      SBRJ      0.8      04         S      *
*   --       ----      ----      -.-      --         -      *
*   ..        ..        ..        .       ..         .      *
*------------------------------------------------------------*
* S_IFUNCT    S_IFTIME   S_IDPCND     S_IDPLOC    S_IDPNO   *
* (Sortie     (Sortie    (Sortie      (Sortie     (Sortie   *
* instrum.    instrum.   instrum.     instrum.    instrum   *
* function)   function   descend      descend     descend   *
*             time)      procedure    procedure   proced.   *
*                        condition)   location)   number)   *
*                                                            *
*    1P         0.4         R           SBBR        02      *
*    IN         0.6         R           SBRJ        01      *
*    IN         0.7         S           SBRJ        03      *
*    --         -.-         -           ----        --      *
*    ..          .          .            ..         ..      *
****************************************************************
```

Figure 27 The Report #1 Individual Flight Record Layout

Report #2 - Mission Type Summary

This report consists of a summary of total times flown per mission type code. At a certain point in time, the Brazilian Air Force has only one strategic operational staff controller. This report is printed for the BAF operational staff controller on a monthly or periodic basis. The report layout is shown in Figure 28.

```
***********************************************************
*                                                         *
* BRAZILIAN AIR FORCE                  R_DATE : 02/01/85  *
*                                      (Report Date)      *
* FLYING UNIT OPERATIONAL CONTROL SYS  R_TIME : 00:00:00  *
*                                      (Report Time)      *
* F_UNITNO: 2732-1353                  R_SDATE: 01/01/85  *
* (Flying Unit Number)                 (Report start date)*
*                                      R_EDATE: 01/31/85  *
* (Report Number) R_NO: 02             (Report end date)  *
*                                                         *
* R_NAME: MISSION TYPE SUMMARY                            *
* (Report Name)                                           *
*  -------------------------------------------------------*
*                             |                           *
*    S_MTCODE     T_PMSTIM    |   S_MTCODE    T_PMSTIM    *
*    (Sortie      (Periodic   |   (Sortie     (Periodic   *
*     mission      total      |    mission     total      *
*     type         sortie     |    type        sortie     *
*     code)        time)      |    code)       time)      *
*                             |                           *
*     07FM03        25.7      |    07FM05       13.4      *
*     07FM07        29.3      |    07TG00        2.5      *
*     ------        --.-      |    ------       --.-      *
*     ------        --.-      |    ------       --.-      *
*       ..           .        |      ..          .        *
*                             |                           *
*                             |                           *
*  -------------------------------------------------------*
*  (General total mission sortie time) G_TMSTIM :   70.9  *
*                                                         *
***********************************************************
```

Figure 28 The Report #2 Mission Type Summary Layout

Report #3 - Crewmember's Summary per Aircraft Type

This report consists of a summary of total time flown per crew members in specific types of aircraft. Every flying unit has one operational controller at a certain point in time. This report is printed for the operational controller on a monthly or periodic basis. The report layout is shown in Figure 29.

```
*************************************************************
* BRAZILIAN AIR FORCE                      R_DATE : 02/01/85   *
*                                          (Report Date)       *
* FLYING UNIT OPERATIONAL CONTROL SYS      R_TIME : 00:00:00   *
*                                          (Report Time)       *
* F_UNITNO: 2732-1353                      R_SDATE: 01/01/85   *
* (Flying Unit Number)                     (Report start date) *
*                                          R_EDATE: 01/31/85   *
* (Report Number) R_NO: 03                 (Report end date)   *
*                                                              *
* R_NAME: CREW MEMBER'S SUMMARY PER AIRCRAFT TYPE              *
* (Report Name)                                                *
*                                                              *
*  A_TYPE          C_SSN          C_LNAME      T_PATYPE        *
* (Aircraft        (Crew          (Crew        (Periodic       *
*  type)           social         member       total           *
*                  security       last         acft. type      *
*                  number)        name)        sortie time)    *
*                                                              *
*  F-103B       291-80-1970       Cunha          10.7          *
*  F-103E       137-69-1976       John           23.4          *
*  F-105F       297-80-1971       Byrd           17.8          *
*  -----        -----------       ----           --.-          *
*    ..            ....            ..             ..           *
*                                                              *
*  T_SQCTIME        T_PCREW       T_YCSTIM       G_TCSTIM      *
*  (Squadron        (Periodic     (Yearly        (General      *
*  total crew        crew         total          total crew    *
*  time)             total)       crew time)     time)         *
*                                                              *
*    225.7            43.2         103.9           675.3       *
*    132.4            61.1          97.5           386.4       *
*    703.2            68.0         168.9          1034.8       *
*    ---.-            --.-         ---.-          ----.-       *
*     ...              ..           ..              ...        *
*************************************************************
```

Figure 29 The Rep #3 Crew Member's Summary per Acft Layout

Report #4 - Missions Summary per Aircraft Number

This is another type of report for the flying unit operational controller. It is not based on individuals mission sorties. Instead, it is a summary of mission sortie types printed on a monthly or periodic basis. The report layout is shown in Figure 30.

```
*************************************************************
*                                                           *
* BRAZILIAN AIR FORCE                  R_DATE : 02/01/85    *
*                                      (Report Date)        *
* FLYING UNIT OPERATIONAL CONTROL SYS  R_TIME : 00:00:00    *
*                                      (Report Time)        *
* F_UNITNO: 2732-1353                  R_SDATE: 01/01/85    *
* (Flying Unit Number)                 (Report start date)  *
*                                      R_EDATE: 01/31/85    *
* (Report Number) R_NO: 04             (Report end date)    *
* R_NAME: MISSIONS SUMMARY PER AIRCRAFT NUMBER              *
* (Report Name)                                             *
*                                                           *
*-----------------------------------------------------------*
* A_TYPE     A_NO      S_MTCODE     T_PMANO      T_YMATIM   *
* (Acft      (Acft     (Sortie      (Periodic    (Yearly    *
* Type)      Number)   Mission      Total Acft   Total Acft *
*                      Type Code)   Mission      Mission    *
*                                   Time)        Time)      *
*                                                           *
* F-103B     2120       07FM03        112.7        342.4    *
* F-103E     2134       07FM05        217.9        287.1    *
* F-105F     2257       07TG00         87.5        305.0    *
* -----      ----       ------        ---.-        ---.-    *
*  ...        ..         ..            ..           ..      *
*                                                           *
*                                                           *
*-----------------------------------------------------------*
*                                                           *
* (Total Periodic Totals Mission Acft No) T_TPMANO: 418.1   *
*                                                           *
* (Total Yearly Totals Mission Acft No)   T_YMTANO: 934.5   *
*                                                           *
*                                                           *
*************************************************************
```

Figure 30 The Rep #4 Missions Summary per Acft Number Layout

Report #5 - Missions Summary per Administrative Unit

This report consists of a summary of crew members mission times per administrative organization and is printed for the administrative units managers on a monthly or periodic basis. Each crew member belongs only to one administrative unit at a certain point in time. The report layout is shown in Figure 31.

```
***********************************************************
*                                                         *
* BRAZILIAN AIR FORCE                 R_DATE : 02/01/85   *
*                                     (Report Date)       *
* FLYING UNIT OPERATIONAL CONTROL SYS R_TIME : 00:00:00   *
*                                     (Report Time)       *
* F_UNITNO: 2732-1353                 R_SDATE: 01/01/85   *
* (Flying Unit Number)                (Report start date) *
*                                     R_EDATE: 01/31/85   *
* (Report Number) R_NO: 05            (Report end date)   *
*                                                         *
* R_NAME: MISSIONS SUMMARY PER ADMINISTRATIVE UNIT        *
* (Report Name)                                           *
*________________________________________________________ *
*                                                         *
* A_TYPE    C_ADUNIT  AD_UCTOT      AD_UPTIM    AD_UYTIM  *
* (Acft     (Acft     (Periodic     (Periodic   (Yearly   *
* Type)     Number)   Adm. Unit.    Adm. Unit.  Adm. Unit.*
*                     Crew members  Sortie      Sortie    *
*                     Totals)       Times)      Times)    *
*                                                         *
* F-103B     AFIT       732.8        112.7         42.4   *
* F-103E    WPAFB       320.3        217.9         87.1   *
* F-105F     AFLC       413.7         87.5         45.0   *
* -----     ----        ------       ---.-        ---.-   *
*  ...       ..           ..           ..           ..    *
*                                                         *
*                                                         *
*                                                         *
***********************************************************
```

Figure 31 The Rep #5 Missions Summary per Adm Unit Layout

Report #6 - Mission Orders Numbers List

This report consists of a list of mission orders numbers performed during the month in ascending order. It is another type of report for the flying unit operational controller but it needs to be consulted many times a day. The report layout is shown in Figure 32.

```
***********************************************************
* BRAZILIAN AIR FORCE                   R_DATE : 02/01/85   *
*                                       (Report Date)       *
* FLYING UNIT OPERATIONAL CONTROL SYS R_TIME : 00:00:00     *
*                                       (Report Time)       *
* F_UNITNO: 2732-1353                   R_SDATE: 01/01/85   *
* (Flying Unit Number)                  (Report start date) *
*                                       R_EDATE: 01/31/85   *
* (Report Number) R_NO: 06              (Report end date)   *
*                                                           *
* R_NAME: MISSIONS ORDERS NUMBERS LIST                      *
* (Report Name)                                             *
*                                                           *
*________________________________________________________  *
*        S_CODENO          A_TYPE          A_NO             *
*        (Sortie           (Aircraft       (Aircraft        *
*         Code              Type)           Number)         *
*         Number)                                           *
*                                                           *
*        104,08701         F-103B           2120            *
*        104,08702         F-103E           2120            *
*        104,08901         F-105F           2123            *
*        ---------         ------           ----            *
*          ....              ..              ..             *
*________________________________________________________  *
*     C_LNAME     S_DEPLOC      S_ARRLOC      S_TIME        *
*     (Crew       (Sortie       (Sortie       (Sortie       *
*      member     Departure     Arrival        Time)        *
*      last       Location)     Location)                   *
*      name)                                                *
*                                                           *
*      Cunha        SBRJ          SBBR          1.5         *
*      John         SBBR          SBRJ          1.6         *
*      Byrd         SBRJ          SBRJ          0.8         *
*      ----         ----          ----          -.-         *
*       ..           ..            ..            .          *
***********************************************************
```

Figure 32 The Report #6 Missions Orders List Layout

Report 7 - Consumed Items per Mission

This report consists of a list of consumed items and quantities per sortie mission type code. Every flying unit has just one flying unit material controller at a certain point in time. This report is printed for the material controller on a monthly or periodic basis. The report layout is shown in Figure 33.

```
*************************************************************
* BRAZILIAN AIR FORCE                   R_DATE : 02/01/85    *
*                                       (Report Date)        *
* FLYING UNIT OPERATIONAL CONTROL SYS   R_TIME : 00:00:00    *
*                                       (Report Time)        *
* F_UNITNO: 2732-1353                   R_SDATE: 01/01/85    *
* (Flying Unit Number)                  (Report start date)  *
*                                       R_EDATE: 01/31/85    *
* (Report Number) R_NO: 07              (Report end date)    *
* R_NAME: CONSUMED ITEMS                                     *
* (Report Name)                                              *
*                                                            *
*  S_MTCODE        A_TYPE        A_NO           S_DEPLOC     *
*  (Sortie        (Aircraft     (Aircraft       (Sortie      *
*   Mission        Type)         Number)        Departure    *
*   Type                                        Location)    *
*   Code)                                                    *
*   07FM03         F-103B         2120            SBRJ       *
*   07FM03         F-103E         2120            SBBR       *
*   07TG00         F-105F         2123            SBRJ       *
*     ..             ..            ..              ..        *
*                                                            *
*   S_ARRLOC       S_TIME        S_ITCODE       S_ITCOQY     *
*   (Sortie       (Sortie        (Sortie        (Sortie      *
*   Arrival        Time)         Item           Item         *
*   Location)                    Code)          Consumed     *
*                                               Quantity)    *
*    SBBR          1.5            MISS77         02          *
*    SBRJ          1.6            ROCK26         04          *
*    SBRJ          0.8            BOMB15         08          *
*     ..            .              ..            ..          *
*                                                            *
*(Periodic Total Time Mission Code Type) T_TPSTIM:   218.1   *
*(Periodic Total Item Consumed Quantity) T_PICOQY:   14      *
*************************************************************
```

Figure 33 The Report #7 Consumed Items Layout

Report 8 - Aircraft Numbers Summary Totals

This is another type of report for the flying unit controller. It consists of a summary of total mission times per aircraft tail numbers. This report is printed for the flying unit material controller on a monthly or periodic basis. The report layout is shown in Figure 34.

```
***********************************************************
* BRAZILIAN AIR FORCE                R_DATE : 02/01/85     *
*                                    (Report Date)         *
* FLYING UNIT OPERATIONAL CONTROL SYS R_TIME : 00:00:00    *
*                                    (Report Time)         *
* F_UNITNO: 2732-1353                R_SDATE: 01/01/85     *
* (Flying Unit Number)               (Report start date)   *
*                                    R_EDATE: 01/31/85     *
* (Report Number) R_NO: 08           (Report end date)     *
*                                                          *
* R_NAME: AIRCRAFT NUMBERS SUMMARY TOTALS                  *
* (Report Name)                                            *
*                                                          *
*  ________________________________________________________ *
*    A_TYPE        A_NO         T-PMSTIM      T_PALNDN     *
*   (Aircraft     (Aircraft     (Periodic     (Periodic    *
*    Type)         Number)       Total         Total       *
*                                Mission       Aircraft    *
*                                Code          Landings)   *
*                                Type)                     *
*    F-103B        2120          27.7          39          *
*    F-103E        2120          31.0          40          *
*    F-105F        2123          12.1          12          *
*    ------        ----          --.-          --          *
*      ..           ..            ..           ..          *
*  ________________________________________________________ *
*                                                          *
*    T_YANO        T_YALNDN      T_CELANO      T_CELALN    *
*    (Yearly       (Yearly       (Total        (Total      *
*    Total         Total         Cell          Cell        *
*    Aircraft      Aircraft      Aircraft      Aircraft    *
*    Time)         Landings)     Time)         Landings)   *
*    137.3          89          1,058.2         1534       *
*    201.0         176          1,321.9         1115       *
*    199.3         127          1,359.7         1307       *
*    ---.-         ---          -----.-         ----       *
*     ...           .             ...            ..        *
***********************************************************
```

Figure 34 The Rep #8 Aircraft Numbers Summary Totals Layout

Report 9 - Aircraft Numbers Status

This report consists of the availability status per aircraft tail number. Each aircraft number can have just one availability status at a certain period of time. There are eight basic defined aircraft availability status. Seven for unavailability, codified from situation A (SITA) throught situation G (SITG), and one for availability codified available (AVAL). This report is printed for the flying unit material controller on a monthly or periodic basis. The report layout is shown in Figure 35.

```
*************************************************************
* BRAZILIAN AIR FORCE                  R_DATE : 02/01/85    *
*                                      (Report Date)        *
* FLYING UNIT OPERATIONAL CONTROL SYS  R_TIME : 00:00:00    *
*                                      (Report Time)        *
* F_UNITNO: 2732-1353                  R_SDATE: 01/01/85    *
* (Flying Unit Number)                 (Report start date)  *
*                                      R_EDATE: 01/31/85    *
* (Report Number) R_NO: 09             (Report end date)    *
* R_NAME: AIRCRAFT NUMBERS STATUS                           *
* (Report Name)                                             *
* _________________________________________________________ *
* A_TYPE     A_NO       A_NPSITA     A_NPSITB     A_NPSITC  *
* (Acft      (Acft      (Acft no.    (Acft no.    (Acft no. *
* Type)      Number)    of period    of period    of period *
*                       Situat. A)   Situat. B)   Situat. C)*
* F-103B      2120        12           07           03      *
* F-103E      2120        00           01           05      *
* F-105F      2123        21           00           00      *
*   ..         ..         ..           ..           ..      *
* _________________________________________________________ *
* A_NPSITD  A_NPSITE   A_NPSITF     A_NPSITG     A_NPAVAL   *
* (Acft no. (Acft no.  (Acft no.    (Acft no.    (Acft no.  *
* of period of period  of period    of period    of period  *
* Stuat. D) Situat. E) Situat. F)   Situat. G)   Available) *
*    01        04         02           07           03      *
*    00        00         00           01           05      *
*    07        02         01           00           00      *
*    ..        ..         ..           ..           ..      *
*************************************************************
```

Figure 35 The Report #9 Aircraft Numbers Status Layout

Report 10 - Consumed Items per Aircraft

This is another type of report for the flying unit material controller. It consists of the quantity of items consumed per aircraft tail numbers and is printed for the material controller on a monthly or periodic basis. This report layout is shown in Figure 36.

```
***********************************************************
*                                                         *
* BRAZILIAN AIR FORCE                  R_DATE : 02/01/85   *
*                                      (Report Date)      *
* FLYING UNIT OPERATIONAL CONTROL SYS R_TIME : 00:00:00   *
*                                      (Report Time)      *
* F_UNITNO: 2732-1353                  R_SDATE: 01/01/85   *
* (Flying Unit Number)                 (Report start date) *
*                                      R_EDATE: 01/31/85   *
* (Report Number) R_NO: 10             (Report end date)   *
*                                                         *
* R_NAME: CONSUMED ITEM PER AIRCRAFT                      *
* (Report Name)                                           *
*                                                         *
*---------------------------------------------------------*
* A_TYPE       A_NO        S_ITNAME   T_PICOQY    S_ITUNIT *
* (Aircraft    (Aircraft   (Sortie    (Periodic   (Sortie  *
* Type)        Number)      Item      Total       Unit     *
*                           Name)     Consumed    Measure) *
*                                     Item)                *
*                                                         *
* F-103B        2120        MISSIL       19          EA    *
* F-103E        2120        ROCKET       12          EA    *
* F-105F        2123        BOMB         04          EA    *
* ------        ----         --          --          --    *
*   ..           ..          ..          ..          ..    *
*                                                         *
*                                                         *
*                                                         *
*                                                         *
*                                                         *
***********************************************************
```

Figure 36 The Report #10 Consumed Items per Aircraft Layout

Report 11 - Consumed Items Quantity

This report consists of the quantity of consumed items per aircraft tail numbers and suppliers for each consumed item. It is printed for the flying unit material controller on a monthly or periodic basis. This report layout is shown in Figure 37.

```
***************************************************************
* BRAZILIAN AIR FORCE                    R_DATE : 02/01/85    *
*                                        (Report Date)        *
* FLYING UNIT OPERATIONAL CONTROL SYS R_TIME : 00:00:00      *
*                                        (Report Time)        *
* F_UNITNO: 2732-1353                    R_SDATE: 01/01/85    *
* (Flying Unit Number)                   (Report start date)  *
*                                        R_EDATE: 01/31/85    *
* (Report Number) R_NO: 11               (Report end date)    *
*                                                             *
* R_NAME: CONSUMED ITEMS QUANTITY                             *
* (Report Name)                                               *
*                                                             *
* S_ITNAME    S_ITSUPP    S_DEPLOC    S_DEPDAT    S_DEPTIM    *
* (Sortie     (Sortie     (Sortie     (Sortie     (Sortie     *
* Item        Item        Departure   Departure   Departure*
* Name)       Supplier)   Location)   Date)       Time)       *
*                                                             *
* MISSIL       IMBEL        SBRJ      01/02/85    10:05:00    *
* ROCKET       TAURUS       SBBR      01/05/85    13:15:39    *
* BOMB         IMBEL        SBBR      01/17/85    08:55:00    *
* ------       ------       ----      --/--/--    --:--:--    *
*   ..           ..          ..          ..          ..       *
*                                                             *
*   A_TYPE          A_NO           S_ITCOQY      S_ITRCNO     *
* (Aircraft       (Aircraft        (Sortie       (Sortie      *
*   Type)          Number)         Item          Item         *
*                                  Consumed      Recept       *
*                                  Quantity)     Number)      *
*                                                             *
*   F-103B          2120            21           123456789    *
*   F-103E          2120            63           234567891    *
*   F-105F          2123            69           456789234    *
*   ------          ----            --           ---------    *
*     ..             ..             ..              ...       *
***************************************************************
```

Figure 37 The Report #11 Consumed Items Quantity Layout

Report 12 - Crew Member's Summary

This is another type of report for the flying unit operational controller. It consists of a list of current data about each crew member. This report layout is printed for the operational controller on a monthly or periodic basis and its layout is shown in Figure 38.

```
*************************************************************
* BRAZILIAN AIR FORCE                    R_DATE : 02/01/85   *
*                                        (Report Date)       *
* FLYING UNIT OPERATIONAL CONTROL SYS R_TIME : 00:00:00       *
*                                        (Report Time)       *
* F_UNITNO: 2732-1353                    R_SDATE: 01/01/85   *
* (Flying Unit Number)                   (Report start date) *
*                                        R_EDATE: 01/31/85   *
* (Report Number) R_NO: 12               (Report end date)   *
* R_NAME: CREW MEMBER'S SUMMARY                              *
* (Report Name)                                              *
*                                                            *
*   C_SSN           C_LNAME              C_CNAME             *
* (Crew member     (Crew                                     *
*  Social Sec.      Last    (Crew Member Complement Name)    *
*  Name)            Name)                                    *
*  291-80-1970      CUNHA       ADILSON MARQUES DA           *
*  219-76-1976      KNODE       DAVID FITZPATRICK            *
*  139-67-1457      ALI         KALIFA YASSER                *
*  -----------      ------      ------------------           *
*                                                            *
*     ...            ..              .....                   *
*                                                            *
*  C_RANK        C_SPEC         C_FQCODE         T_PCXFUN    *
*  (Crew         (Crew          (Crew            (Periodic   *
*  Mermber       Member         Function         Crew Total  *
*  Rank)         Speciality)    Qualification    Executed    *
*                               Code)            Function)   *
*    MAJ           PILOT          1P               25.8      *
*    CAPT          NAVIGT         IN               12.7      *
*    COL           FLYENG         IN               35.0      *
*    --            ------         --               --.-      *
*    ..            ..             ..               ..        *
*************************************************************
```

Figure 38 The Report #12 Crew Members' Summary Layout

Report 13 - Crew member Totals

This report consists of crew member's total sortie times and landings. It complements report 1, is generated one for each crew member on a monthly or periodic basis, and its layout is shown in Figure 39.

```
***********************************************************
* BRAZILIAN AIR FORCE                    R_DATE : 02/01/85 *
* FLYING UNIT OPERATIONAL CONTROL SYS    (Report Date)     *
* F_UNITNO: 2732-1353                    R_TIME : 00:00:00 *
* (Flying Unit Number)                   (Report Time)     *
* R_NO : 13                              R_SDATE: 01/01/85 *
* (Report Number)                        (Rep Start Date)  *
* R_NAME: CREW MEMBER INDIVIDUAL TOTALS R_EDATE: 01/31/85 *
* (Report Name)                          (Report End Date) *
*                                                          *
* (Crew Member Social Sec. No.) C_SSN   : 291-80-1970      *
* (Crew Rank, Speciality) C_RANK, C_SPEC: Major, Pilot     *
* (Crew Last Name) C_LNAME              : Cunha            *
* (Crew Complement Name) C_CNAME        : Adilson M. da    *
* ________________________________________________________ *
* T_CF1TIM     T_CF1LND     T_CF2TIM    T_CF2LND    T_CF3TIM *
* (Crew        (Crew        (Crew       (Crew       (Crew    *
* Total        Total        Total       Total       Total    *
* Time in      Landings     Time in     Landings    Time in  *
* Function     in Funct.    Funct.      in Funct.   Function *
* 1)           1)           2)          2)          3)       *
* 2208.8        2876          978.9       1580        187.8  *
* 1087.6         876          432.4        587         63.8  *
* 3876.3        4097         1045.1       1807        780.0  *
* ----.-        ----         ----.-       ----        -----  *
*   ...          ..            ...         ..          ...   *
* ________________________________________________________ *
* T_CF3LND     T_PMSTIM    T_YCSTIM    G_TCSTIM    G_TLNDNO  *
* (Crew        (Periodic   (Yearly     (General    (General  *
* Total        Crew        Crew        Crew        Crew      *
* Landings     Total       Total       Total       Total     *
* in Funct.    Sortie      Sortie      Sortie      Sortie    *
* 3)           Time)       Time)       Time)       Landings) *
*   211         32.2       326.1       3763.9       5003     *
*   110         28.9       298.5       2015.4       2600     *
*   987         12.8       132.1       7076.7       9000     *
*   ---         --.-       ---.-       ----.-       ----     *
*    .           ..         ...         ...          ..      *
***********************************************************
```

Figure 39 The Report #13 Crew Member Totals Layout

Report 14 - Crew member's Totals Summary

This report consists of a summary of crew member's total sortie times and landings. Each flying unit has just one tactical operational controller at certain point in time. This report is printed for the tactical operational controller on a monthly or periodic basis and its layout is shown in Figure 40.

```
*************************************************************
* BRAZILIAN AIR FORCE                      R_DATE : 02/01/85   *
*                                          (Report Date)       *
* FLYING UNIT OPERATIONAL CONTROL SYS R_TIME : 00:00:00   *
*                                          (Report Time)       *
* F_UNITNO: 2732-1353                      R_SDATE: 01/01/85   *
* (Flying Unit Number)                     (Report start date) *
*                                          R_EDATE: 01/31/85   *
* (Report Number) R_NO: 14                 (Report end date)   *
* R_NAME: CREW MEMBER'S TOTALS SUMMARY                         *
* (Report Name)                                                *
*______________________________________________________________*
* C_SSN     C_LNAME  T_CF1TIM  T_CF1LND  T_CF2TIM  T_CF2LND *
* (Crew     (Crew    (Crew     (Crew     (Crew     (Crew    *
* Social    Last     Total     Total     Total     Total    *
* Security  Name)    Time in   Landings  Time in   Landings *
*  Name)             Funct1)   Funct1)   Funct2)   Funct2)  *
*291801970  CUNHA   2208.8    2876      978.9     1580     *
*219761976  KNODE   1087.6     876      432.4      587     *
*139671457  ALI     3876.3    4097     1045.1     1807     *
*---------  -----   ----.-    ----     ----.-     ----     *
*  .....     ...     ...       ..       ...        ..      *
*______________________________________________________________*
*T_CF3TIM  T_CF3LND  T_PMSTIM T_YCSTIM G_TCSTIM  G_TLNDNO *
* (Crew     (Crew    (Periodic (Yearly  (General  (General *
* Total     Total     Crew      Crew     Crew      Crew     *
* Time in   Landings  Total     Total    Total     Total    *
* Funct.    in Funct. Sortie    Sortie   Sortie    Sortie   *
* 3)        3)        Time)     Time)    Time)     Landings)*
*  187.8    211        32.2     326.1   3763.9     5003    *
*   63.8    110        28.9     298.5   2015.4     2600    *
*  780.0    987        12.8     132.1   7076.7     9000    *
*  ---.-    ---        --.-     ---.-   ----.-     ----    *
*   ...      .          ..       ...     ...        ..     *
*************************************************************
```

Figure 40 The Rep #14 Crew Members' Totals Summary Layout

Appendix B

The BAF FLUNITOC System Data Dictionary

After analyzing individually each current file management system report, the following data dictionary began to be constructed with the referenced data items in alphabetical order (1).

ACCIDENT	A system entity defined as an unexpected and undesirable event that can happen involving crew members, aircraft, and mission sortie types. It is not important for the FLUNITOC system but for future system developments.
ACCIDINV	The system entity performed as a consequense of an accident. It is also important for future system developments.
ACCIDPRV	The sytem entity performed in order to prevent accidents, that is, before accident occurrencies. It is also important for future developments.
ADMSUPPO	Another system entity that refers to any admnistrative event to support the flying activities. It is also important for future developments.
A_AVSITU	The aircraft availability situation or status code, defined as seven inoperative status codes and one operative or available status code. For example, available = av, situationa = stata (for stataus a), situationb = statb, and so on.
A_AVDATE	The date system entity also called aircraft available date.
A_AVTIME	The time system entity also called aircraft available time.
A-AVPERD	The aircraft availability period, in other words, a system window of eight consecutive hours period, in which an aircraft stands on a specific availability situation or status.

AD_UCTOT	The administrative unit crew members' total.
AD_UPTIM	The administrative unit periodic sortie time.
AD_UYTIM	The administrative unit yearly sortie time.
A_NO	The aircraft system entity also called aircraft tail number or simple aircraft number.
A_NPSTAA	The aircraft number of periods on status A.
A_NPSTAB	The aircraft number of periods on status B.
A_NPSTAC	The aircraft number of periods on status C.
A_NPSTAD	The aircraft number of periods on status D.
A_NPSTAE	The aircraft number of periods on status E.
A_NPSTAF	The aircraft number of periods on status F.
A_NPSTAG	The aircraft number of periods on status G.
A_NPAVAL	The aircraft number of periods operative or available.
A_STANPR	The aircraft status number of periods, defined as the number of periods that an aircraft stands on a specific status code.
A_TYPE	An aircraft type.
C_ADUNIT	The system entity representing the crew members' administrative unit.
C_CNAME	A crew member complement name (all name but the last name).
C_FQCODE	The system entity representing the crew member functional qualification code.
C_LNAME	A crew member last name.
C_RANK	A crew member rank.
CREWMEMB	The crew member system entity, self explanatory.
C_SPEC	A crew member speciality.
C_SSN	The crew member system entity also called crew

member social security number.

D_YEAR Part of the date system entity corresponding to the referenced year (YY).

FLYSAFOF A system entity that stands for flight safety officer, the person who investigates and prevents flying accidents.

F_UNITNO The flying unit system entity that represents a flying unit code number, defined as a sequence of four pairs of numbers. For instance when the F_UNITNO = 02231207, the first pair 02 means the numbered air force two, the second pair 23 means the twenty-third wing, the third pair 12 means the twelfth group, and the last pair 07 means the seventh squadron. Thus, the complete code number means the seventh squadron of the twelfth group of the twenty-third wing of the second air force.

F_UNITRG The flying unit regions where flying units are assigned to fly.

G_TCSTIM The general total of crew mumber sortie time.

G_TLNDNO The general total landing number per sorties.

G_TMSTIM The general total of missions sortie time.

INSTRUCT A system entity that refers to instructors, important only for future developments.

MAINTPER A system entity that refers to maintenance personnel, in other words, the personnel responsable for supporting aircraft maintenance, important only for future system developments.

MAINTMAT A system entity that refers to maintenance material, the material for supporting aircraft maintenance, important only for futute system developments.

MAINTSRV A sytem entity that refers to maintenance service, in other words, services executed in aircraft by maintenaince personnel using maintenance material. It is important only for future system developments.

OPERPLAN A system entity that refers to an operational planning, important only for future system developments.

R_DATE	The date system entity also called report date (YY/MM/DD).
R_EDATE	The date system entity also called report ending date (YY/MM/DD).
R_NAME	A report name.
R_NO	The report system entity also called report number.
R_SDATE	The date system entity also called report starting period date (YY/MM/DD).
R_TIME	The time system entity also called report generation time (HH:MM:SS).
S_ARRLOC	A sortie arrival location code.
S_CODENO	The system entity representing the sortie code number and defined as a sequence of eight digit numbers, where the first six digits codify the mission order number and the last two digits codify the sortie number within the mission number. For instance, when the S_CODENO = 16532403, it means the mission order number 165324 sortie number 03, in other words, the third sortie of the mission order number 165324.
S_DEPDAT	The date system entity also called sortie departure date.
S_DEPLOC	A sortie departure location code.
S_DEPTIM	The time system entity also called sortie departure time.
S_IDPCON	The instrument descending procedure condition in a sortie.
S_IDPLOC	An instrument descending procedure location code.
S_IDPNO	The number of instrument descending procedures in a sortie.
S_IFCOND	The instrument function condition in a sortie.
S_IFTIME	The instrument function time in a sortie.

S_IFUNCT	An instrument function type in a sortie.
S_ITCODE	A sortie item code.
S_ITCOQY	The sortie item consumed quantity.
S_ITNAME	A sortie item name.
S_ITSUPP	A sortie item supplier.
S_ITRCNO	The consumed item system entity also called sortie item receipt number.
S_ITUNIT	A sortie item unit of measure.
S_LANDNO	The number of landings in a sortie.
S_MTCODE	The sortie mission type code of a M_ORDNO.
S_TIME	The time system entity also called sortie time.
SUBJECTS	A system entity that refers to the subjects related to the flying activities taken by trainees and taught by instructors, in order to qualify crew members to execute flying unit missions. It is important only for future system developments.
SUPPPERS	A system entity that refers to supporting personnel, the necessary activities for supporting crew members. It is important for future system developments.
SUPPMATR	A system entity that refers to supporting material, the necessary activities for supporting other activities related with the flying unit. It is important for future system developments.
S_XFUNCT	An executed function type in a sortie.
T_CELANO	The total cell time per aircraft number.
T_CELALN	The total cell landing per aircraft number.
T_CF1TIM	The crew member total sortie times executing sortie mission functions type 1, that is, with functional qualification code (C-FQCODE) equal to CH for checker, IN for instructor, FP for first pilot, or ST for student functions.

T_CF1LND	The crew member total landings executing sortie mission functions type 1, that is, with functional qualification code (C-FQCODE) equal to CH for checker, IN for instructor, FP for first pilot, or ST for student functions.
T_CF2TIM	The crew member total sortie times executing sortie mission functions type 2, that is, with functional qualification code (C-FQCODE) equal to SP for second pilot function.
T_CF2LND	The crew member total landings executing sortie mission functions type 2, that is, with functional qualification code (C-FQCODE) equal to SP for second pilot function.
T_CF3TIM	The crew member total sortie times executing sortie mission functions type 3, that is, with functional qualification code (C-FQCODE) equal to NC for navigator checker, NI for navigator instructor, FN for first navigator, or OB for observer functions.
T_CF3LND	The crew member total landings executing sortie mission functions type 3, that is, with functional qualification code (C-FQCODE) equal to NC for navigator checker, NI for navigator instructor, FN for first navigator, or OB for observer functions.
T_MORDNO	The total of sortie time per mission order number.
T_PALNDN	The periodic total of landing numbers per aircraft number.
T_PATYPE	The periodic total aircraft type sortie time.
T_PCREW	The periodic total of crew member sortie time.
T_PCXFUN	The periodic total of a crew member executed function.
T_PICOQY	The periodic total item consumed quantity.
T_PMANO	The periodic total of mission sortie time per aircraft number.
T_PMSTIM	The periodic total of sortie time in a mission code type.
T_SQCTIM	The squadron total sortie time of a crew

member.

T_TPMANO	The total of periodic totals of mission sortie times per aircraft number.
TRAINEES	The system entity that refers to the crew members in process of training. It is important for future system developments.
T_MSTIME	The total mission type code sortie time.
T_YALNDN	The yearly total of landings per aircraft numbers.
T_YANO	The yearly total of sortie times per aircraft numbers.
T_YCSTIM	The yearly total crew member sortie time.
T_YMATIM	The yearly total of mission sortie time per aircraft number.
T_YMTANO	The total yearly totals of mission sortie time per aircraft number.

Appendix C

The FLUNITOC System Third Normal Form (3NF) Relations

To develop the third normal form relations for each report, first the set of data items within each report was identified. After that, the relationships between data items were determined and described by identifying the key data items and nonkey data items for each relation. Finally, third normal form relations for each set of data items where derived. Where this was not possible for individual reports, the data from reports were merged to establish the third normal form relations.

Report 1 - Individual Flight Record

a) The data items representing the entities of this report are: F-UNITNO, R-NO, R-NAME, C-SSN, C-RANK, C-SPEC, C-LNAME, C-CNAME, S-DEPDAT, S-CODENO, S-MTCODE, A-TYPE, A-NO, S-XFUNCT, S-DEPLOC, S-ARRLOC, S-TIME, S-LANDNO, S-ICOND, S-IFUNCT, S-IFTIME, S-IDPCND, S-IDPLOC, and S-IDPNO.

b) The relationships between the data items of this report are:

(1) R-NO <----------> R-NAME, that means, for a given report number (R-NO) there is only one report name (R-NAME), that is, a one-to-one mapping represented as <-------> a two opposite

arrows;

(2) R-NO,R-DATE,R-TIME <----------> R-SDATE, R-EDATE, that means, for a given report number (R-NO) with report date (R-DATE) and report time (R-TIME) there is only one report starting date (R-SDATE) and report ending date (R-EDATE), that is, a one-to-one mapping;

(3) F-UNITNO <--<-------> C-SSN, A-NO, that means, for a given flying unit number (F-UNITNO), there may be many crew members social security numbers (C_SSN) and aircraft numbers (A-NO), that is, a one-to-many mapping, represented as <--<------>;

(4) S-CODENO <--<-------> S-MTCODE, that is, for a given sortie code number (S-CODENO), there may be many sortie mission type code (S-MTCODE);

(5) F-UNITNO,S-CODENO <--<-------> S-MTCODE, S-DEPLOC, S-ARRLOC, S-TIME, S-LANDNO, A-NO, C-SSN, that is, for a given flying unit number (F_UNITNO) with sortie code number (S-CODENO) there may be many sortie mission type code (S-MTCODE), sortie departure location (S-DEPLOC), sortie arrival location (S-ARRLOC), sortie time (S-TIME), sortie landing number (S-LANDNO), aircraft number (A-NO), and crew member social security number (C-SSN);

(6) C-SSN <--<-------> C-RANK, C-SPEC, C-LNAME, C-CNAME, C-FQCODE, that is, for a given crew

member social security number (C-SSN) there may be many crew member rank (C-RANK), crew speciality (C-SPEC), crew last name (C-LNAME), crew complement name (C-CNAME), and crew functional qualification code (C-FQCODE);

(7) C-FQCODE <--<-------> S-XFUNCT, that is, for a given crew member functional qualification code (C-FQCODE) there may be many crew members executed functions (C-XFUNCT);

(8) A-NO <--<-------> A-TYPE, that is, for a given aircraft number (A-NO) there may be many aircraft type (A-TYPE); and

(9) S-CODENO,S-DEPDAT,A-ANO <--<-------> C-SSN, S-XFUNC, S-IFUNCT, S-IFTIM, S-IFCOND, S-IDPLOC, S-IDPCON, S-IDPNO, that is, for a give sortie code number (S-CODENO) with sortie departure date (S-DEPDAT) and also with aircraft number (A-NO) there may be many crew member social security number (C-SSN), sortie executed function (S-XFUNCT), sortie instrument function (S-IFUNCT), sortie instrument function time (S-IFTIM), sortie instrument function condition (S-IFCOND), sortie instrument descend procedure location (S-IDPLOC), sortie instrument descend procedure condition (S-IDPCON), and sortie instrument descend procedure number (S-IDPNO).

c) The third normal form relations for the Individual Flight Record Report are:

(1) R-NO <----------> R-NAME;

(2) R-NO,R-DATE,R-TIME <----------> R-SDATE, R-EDATE;

(3) F-UNITNO <--<-------> C-SSN, A-NO;

(4) S-CODENO <--<-------> S-MTCODE;

(5) F-UNITNO,S-CODENO <--<-------> S-DEPLOC, S-ARRLOC, S-TIME, S-LANDNO;

(6) C-SSN <--<-------> C-RANK, C-SPEC, C-LNAME, C-CNAME, C-FQCODE;

(7) C-FQCODE <--<-------> S-XFUNCT;

(8) A-NO <--<-------> A-TYPE; and

(9) S-CODENO,S-DEPDAT,A-ANO <--<-------> C-SSN, S-XFUNCT, S-IFUNCT, S-IFTIM, S-IFCOND, S-IDPLOC, S-IDPCON, S-IDPNO.

Report 2 - Mission Type Summary

a) The data items representing the entities of this report are: F-UNITNO, R-NO, R-NAME, R-DATE, R-TIME, R-SDATE, R-EDATE, S-MTCODE, T-PMSTIM, and G-TMSTIM.

b) The relationships between the data items of this report are:

(10) R-NO <----------> R-NAME, that means, for a given report number (R-NO) there is only one report name (R-NAME), that is, a one-to-one

mapping represented as <-------> a two opposite arrows;

(11) R-NO,R-DATE,R-TIME <----------> R-SDATE, R-EDATE, that means, for a given report number (R-NO) with report date (R-DATE) and report time (R-TIME) there is only one report starting date (R-SDATE) and report ending date (R-EDATE), that is, a one-to-one mapping;

(12) F-UNITNO,S-MTCODE <--------> G-TMSTIM, that is, for a given flying unit number (F-UNITNO) with sortie mission type code (S-MTCODE), there is only one general total of mission sortie time (G-TMSTIM); and

(13) F-UNITNO,S-MTCODE,R-SDATE,R-EDATE <--<----> T-PMSTIM, that is, for a flying unit number (F-UNITNO) with sortie mission type code (S-MTCODE), report starting date (R-SDATE), and report ending date (R-EDATE), there may be many periodic totals of sortie times (T-PMSTIM).

c) The third normal form relations for the Mission Type Summary Report are:

(10) R-NO <-------> R-NAME;

(11) R-NO,R-DATE,R-TIME <-------> R-SDATE, R-EDATE;

(12) F-UNITNO,S-MTCODE <--------> G-TMSTIM; and

(13) F-UNITNO,S-MTCODE,R-SDATE,R-EDATE <--<----> T-PMSTIM.

Report 3 - Crew Member's Summary per Aircraft Type

a) The data items representing the entities of this report are: F-UNITNO, R-NO, R-NAME, R-DATE, R-TIME, R-SDATE, R-EDATE, A-TYPE, C-SSN, C-LNAME, T-PATYPE, T-SQCTIM, T-PCREW, T-YCSTIM, and G-TCSTIM.

b) The relationships between the data items of this report are:

(14) R-NO <----------> R-NAME, that means, for a given report number (R-NO) there is only one report name (R-NAME), that is, a one-to-one mapping represented as <-------> a two opposite arrows;

(15) R-NO,R-DATE,R-TIME <----------> R-SDATE, R-EDATE, that means, for a given report number (R-NO) with report date (R-DATE) and report time (R-TIME) there is only one report starting date (R-SDATE) and report ending date (R-EDATE), that is, a one-to-one mapping;

(16) F-UNITNO,A-TYPE,C-SSN <-----> G-TCSTIM, T-PATYPE, T-SQCTIM, that is, for a given flying unit number (F-UNITNO) with aircraft type (A-TYPE) and crew social security number (C-SSN) there is only one general total crew sortie time (G-TCSTIM), total periodic aircraft type (T-PATYPE), and total squadron crew time

(T-SQCTIM);

(17) C-SSN <-----> C-LNAME, that is, for a given crew member social security number (C-SSN) there is only one crew last name (C-LNAME);

(18) F-UNITNO,A-TYPE,C-SSN,D-YEAR <-----> T-YCSTIM, that is, for a given flying unit number (F-UNITNO) with aircraft type (A-TYPE), crew social security number (C-SSN), and data year reference (D-YEAR) there is only one yearly total crew member time (T-YCSTIM); and

(19) F-UNITNO,A-TYPE,C-SSN,R-SDATE,R-EDATE <-----> T-PCREW, that is, for a given flying unit number (F-UNITNO) with aircraft type (A-TYPE), crew social security number (C-SSN), report starting date (R-SDATE), and report ending date (R-EDATE), there is only one periodic total of crew member sortie time (T-PCREW).

c) The third normal form relations for the Crew Member's Summary per Aircraft Type Report are:

(14) R-NO <-----> R-NAME;

(15) R-NO,R-DATE,R-TIME <-----> R-SDATE, R-EDATE;

(16) F-UNITNO,A-TYPE,C-SSN <-----> G-TCSTIM, T-PATYPE, T-SQCTIM;

(17) C-SSN <-----> C-LNAME;

(18) F-UNITNO,A-TYPE,C-SSN,D-YEAR <-----> T-YCSTIM; and

(19) F-UNITNO,A-TYPE,C-SSN,R-SDATE,R-EDATE <-----> T-PCREW.

Report 4 - Missions Summary per Aircraft Number

a) The data items representing the entities of this report are: F-UNITNO, R-NO, R-NAME, R-DATE, R-TIME, R-SDATE, R-EDATE, A-TYPE, A-NO, S-MTCODE, T-PMANO, T-YMATIM, T-TPMANO, and T-YMTANO.

b) The relationships between the data items of this report are:

(20) R-NO <-----------> R-NAME, that means, for a given report number (R-NO) there is only one report name (R-NAME), that is, a one-to-one mapping represented as <-------> a two opposite arrows;

(21) R-NO,R-DATE,R-TIME <-----------> R-SDATE, R-EDATE, that means, for a given report number (R-NO) with report date (R-DATE) and report time (R-TIME) there is only one report starting date (R-SDATE) and report ending date (R-EDATE), that is, a one-to-one mapping;

(22) F-UNITNO , that is, the flying unit number by itself;

(23) A-NO <--<-------> A-TYPE, that is, for a given aircraft number (A-NO) there may be many aircraft type (A-TYPE);

(24) A-NO,S-MTCODE,D-YEAR <------> T-YMATIM, T-YMTANO, that is , for a given aircraft number (A-NO) with a sortie mission type code (S-MTCODE) and a data year reference (D-YEAR), there is only one yearly total of mission sortie time per aircraft number (T-YMATIM) and one yearly total of mission sortie time per aircraft number (T-YMTANO); and

(25) A-NO,S-MTCODE,R-SDATE,R-EDATE <------> T-PMANO, T-TPMANO, that is, for a given aircraft number (A-NO) with a sortie mission type code (S-MTCODE), report starting date (R-SDATE) and report ending date (R-EDATE) there is only one periodic total of mission sortie time per aircraft number (T-PMANO) and one total of periodic totals of mission sortie times per aircraft number (T-TPMANO).

c) The third normal form relations for the Missions Summary per Aircraft Number Report are:

(20) R-NO <------> R-NAME;

(21) R-NO,R-DATE,R-TIME <------> R-SDATE, R-EDATE;

(22) F-UNITNO

(23) A-NO <--<---> A-TYPE;

(24) A-NO,S-MTCODE,D-YEAR <------> T-YMATIM, T-YMTANO; and

(25) A-NO,S-MTCODE,R-SDATE,R-EDATE <------> T-PMANO,

T-TPMANO.

Report 5 - Missions Summary per Administrative Unit

a) The data items representing the entities of this report are: F-UNITNO, R-NO, R-NAME, R-DATE, R-TIME, R-SDATE, R-EDATE, A-TYPE, C-ADUNIT, AD-UCTOT, AD-UPTIM, and AD-UYTIM.

b) The relationships between the data items of this report are:

(26) R-NO <----------> R-NAME, that means, for a given report number (R-NO) there is only one report name (R-NAME), that is, a one-to-one mapping represented as <-------> a two opposite arrows;

(27) R-NO,R-DATE,R-TIME <----------> R-SDATE, R-EDATE, that means, for a given report number (R-NO) with report date (R-DATE) and report time (R-TIME) there is only one report starting date (R-SDATE) and report ending date (R-EDATE), that is, a one-to-one mapping;

(28) A-NO <--<-------> A-TYPE, that is, for a given aircraft number (A-NO) there may be many aircraft type (A-TYPE);

(29) F-UNITNO,A-TYPE,C-ADUNIT <------> AD-UCTOT, that is, for a given Flying unit (F-UNITNO), aircraft type (A-TYPE) and Crew member administrative unit (C-ADUNIT), there is only

one administrative unit crew member's total (AD-UCTOT);

(30) A-TYPE,C-ADUNIT,D-YEAR <------> AD-UYTIM, that is, for a given aircraft type (A-TYPE), administrative unit (C-ADUNIT), and data year reference (D-YEAR), there is only one administrative unit yearly sortie total time; and

(31) A-TYPE,C-ADUNIT,R-SDATE,R-EDATE <------> AD-UPTIM, that is, for a given aircraft type (A-TYPE), administrative unit (C-ADUNIT), report starting date (R-SDATE) and report ending date (R-EDATE), there is only one administrative unit periodic sortie time (AD-UPTIM).

c) The third normal form relations for the Missions Summary per Administrative Unit Report are:

(26) R-NO <------> R-NAME;

(27) R-NO,R-DATE,R-TIME <------> R-SDATE, R-EDATE;

(28) A-NO <--<---> A-TYPE;

(29) F-UNITNO,A-TYPE,C-ADUNIT <------> AD-UCTOT;

(30) A-TYPE,C-ADUNIT,D-YEAR <------> AD-UYTIM; and

(31) A-TYPE,C-ADUNIT,R-SDATE,R-EDATE <------> AD-UPTIM.

Report 6 - Missions Orders Numbers List

a) The data items representing the entities of this report

are: F-UNITNO, R-NO, R-NAME, R-DATE, R-TIME, R-SDATE, R-EDATE, S-CODENO, A-TYPE, A-NO, C-LNAME, S-DEPLOC, S-ARRLOC, and S-TIME.

b) The relationships between the data items of this report are:

(32) R-NO <----------> R-NAME, that means, for a given report number (R-NO) there is only one report name (R-NAME), that is, a one-to-one mapping represented as <-------> a two opposite arrows;

(33) R-NO,R-DATE,R-TIME <----------> R-SDATE, R-EDATE, that means, for a given report number (R-NO) with report date (R-DATE) and report time (R-TIME) there is only one report starting date (R-SDATE) and report ending date (R-EDATE), that is, a one-to-one mapping;

(34) A-NO <--<-------> A-TYPE, that is, for a given aircraft number (A-NO) there may be many aircraft type (A-TYPE); and

(35) F-UNITNO,S-CODENO <--<---> A-NO, C-LNAME, S-DEPLOC, S-ARRLOC, S-TIME, that is, for a given flying unit (F-UNITNO) and sortie code number (S-CODENO), that may be many aircraft numbers (A-NO), crew last names (C-LNAME), sortie departure location (S-DEPLOC), sortie arrival location (S-ARRLOC) and sortie time (S-TIME).

c) The third normal form relations for the Missions Orders Numbers List Report are:

(32) R-NO <------> R-NAME;

(33) R-NO,R-DATE,R-TIME <------> R-SDATE, R-EDATE;

(34) A-NO <------> A-TYPE; and

(35) F-UNITNO,S-CODENO <--<---> A-NO, C-LNAME, S-DEPLOC, S-ARRLOC, S-TIME.

Report 7 - Consumed Items per Mission

a) The data items representing the entities of this report are: F-UNITNO, R-NO, R-NAME, R-DATE, R-TIME, R-SDATE, R-EDATE, S-MTCODE, A-TYPE, A-NO, S-DEPLOC, S-ARRLOC, S-TIME, S-ITCODE, S-ITCOQY, T-MSTIME, and T-PICOQY.

b) The relationships between the data items of this report are:

(36) R-NO <----------> R-NAME, that means, for a given report number (R-NO) there is only one report name (R-NAME), that is, a one-to-one mapping represented as <-------> a two opposite arrows;

(37) R-NO,R-DATE,R-TIME <----------> R-SDATE, R-EDATE, that means, for a given report number (R-NO) with report date (R-DATE) and report time (R-TIME) there is only one report starting date (R-SDATE) and report ending date (R-EDATE), that is, a one-to-one mapping;

(38) F-UNITNO <--<-----> A-NO, that is, for a given flying unit (F-UNITNO) there may be many aircraft numbers (A-NO);

(39) A-NO <--<-------> A-TYPE, that is, for a given aircraft number (A-NO) there may be many aircraft type (A-TYPE); and

(40) A-NO,S-MTCODE,R-SDATE,R-EDATE <-------> S-TIME, S-ITCODE, S-ITCOQY, S-DEPLOC, T-MSTIME, T-PICOQY, that is, for a given aircraft number (A-NO), sortie mission type code (S-MTCODE), report starting date (R-SDATE) and report ending date (R-EDATE), there is only one sortie time (S-TIME), sortie item code (S-ITCODE), sortie item consumed quantity (S-ITCOQY), sortie departure location)S_DEPLOC), total mission sortie time (T-MSTIME), and periodic total item consumed quantity (T-PICOQY).

c) The third normal form relations for the Consumed Item per Mission Report are:

(36) R-NO <-------> R-NAME;

(37) R-NO,R-DATE,R-TIME <-------> R-SDATE, R-EDATE;

(38) F-UNITNO <--<-----> A-NO;

(39) A-NO <--<----> A-TYPE; and

(40) A-NO,S-MTCODE,R-SDATE,R-EDATE <-------> S-TIME, S-ITCODE, S-ITCOQY, S-DEPLOC, T-MSTIME, T-PICOQY.

Report 8 - Aircraft Numbers Summary Totals

a) The data items representing the entities of this report are: F-UNITNO, R-NO, R-NAME, R-DATE, R-TIME, R-SDATE, R-EDATE, A-TYPE, A-NO, T-PMSTIM, T-PALNDN, T-YANO, T-YALNDN, T-CELANO, and T-CELALN.

b) The relationships between the data items of this report are:

(41) R-NO <----------> R-NAME, that means, for a given report number (R-NO) there is only one report name (R-NAME), that is, a one-to-one mapping represented as <-------> a two opposite arrows;

(42) R-NO,R-DATE,R-TIME <----------> R-SDATE, R-EDATE, that means, for a given report number (R-NO) with report date (R-DATE) and report time (R-TIME) there is only one report starting date (R-SDATE) and report ending date (R-EDATE), that is, a one-to-one mapping;

(43) F-UNITNO , that is, the flying unit number by itself;

(44) A-NO <--<-------> A-TYPE, T-CELANO, T-CELALN, that is, for a given aircraft number (A-NO) there may be many aircraft type (A-TYPE), total cell times per aircraft number (T-CELANO), and total cell landings per aircraft number (T-CELALN);

(45) A-NO,R-SDATE,R-EDATE <--<------> T-PMSTIM, T-PALNDN, that is, for a given aircraft number (A-NO) with reporting starting date (R-SDATE) and report ending date (R-EDATE), there may be many periodic totals of sortie times (T-PMSTIM) and periodic totals of landing numbers per aircraft numbers (T-PALNDN); and

(46) A-NO,D-YEAR <--<------> T-YANO, T-YALNDN, that is, for a given aircraft number (A-NO) and data year reference (D-YEAR), there may be many yearly total of sortie times per aircraft numbers (T-YANO) and yearly total of landings per aircraft numbers (T-YALNDN).

c) The third normal form relations for the Aircraft Numbers Summary Totals Report are:

(41) R-NO <---------> R-NAME;

(42) R-NO,R-DATE,R-TIME <---------> R-SDATE, R-EDATE;

(43) F-UNITNO

(44) A-NO <--<------> A-TYPE, T-CELANO, T-CELALN;

(45) A-NO,R-SDATE,R-EDATE <--<------> T-PMSTIM, T-PALNDN; and

(46) A-NO,D-YEAR <--<------> T-YANO, T-YALNDN.

Report 9 - Aircraft Numbers Status

a) The data items representing the entities of this report

are: F-UNITNO, R-NO, R-NAME, R-DATE, R-TIME, R-SDATE, R-EDATE, A-TYPE, A-NO, A-NPSITA, A-NPSITB, A-NPSITC, A-NPSITD, A-NPSITE, A-NPSITF, A-NPSITG, and A-NPAVAL.

b) The relationships between the data items of this report are:

(47) R-NO <----------> R-NAME, that means, for a given report number (R-NO) there is only one report name (R-NAME), that is, a one-to-one mapping represented as <-------> a two opposite arrows;

(48) R-NO,R-DATE,R-TIME <----------> R-SDATE, R-EDATE, that means, for a given report number (R-NO) with report date (R-DATE) and report time (R-TIME) there is only one report starting date (R-SDATE) and report ending date (R-EDATE), that is, a one-to-one mapping;

(49) A-NO <--<-------> A-TYPE, that is, for a given aircraft number (A-NO) there may be many aircraft type (A-TYPE); and

(50) A-NO,R-SDATE,R-EDATE <--<---------> A-NPSTAA, A-NPSTAB, A-NPSTAC, A-NPSTAD, A-NPSTAE, A-NPSTAF, A-NPSTAG, A-NPSTAV, that is, for a given aircraft number (A-NO) with reporting starting date (R-SDATE) and report ending date (R-EDATE), there may be many aircraft number of periods with status A (A-NPSTAA), status B

(A-NPSTAB), status C (A-NPSTAC), status D (A-NPSTAD), status E (A-NPSTAE), status F (A-NPSTAF), status G (A-NPSTAG), and status available (A-NPSTAV).

c) The third normal form relations for the Aircraft Numbers Status Report are:

(47) R-NO <------------> R-NAME;

(48) R-NO,R-DATE,R-TIME <------------> R-SDATE, R-EDATE;

(49) A-NO <--<---------> A-TYPE;

(50) A-NO,R-SDATE,R-EDATE <--<---------> A-NPSITA, A-NPSITB, A-NPSITC, A-NPSITD, A-NPSITE, A-NPSITF, A-NPSITG, A-NPAVAL;

Report 10 - Consumed Item per Aircraft

a) The data items representing the entities of this report are: F-UNITNO, R-NO, R-NAME, R-DATE, R-TIME, R-SDATE, R-EDATE, A-TYPE, A-NO, S-ITNAME, T-PICOQY, and S-ITUNIT.

b) The relationships between the data items of this report are:

(51) R-NO <----------> R-NAME, that means, for a given report number (R-NO) there is only one report name (R-NAME), that is, a one-to-one mapping represented as <-------> a two opposite arrows;

(52) R-NO,R-DATE,R-TIME <----------> R-SDATE,

R-EDATE, that means, for a given report number (R-NO) with report date (R-DATE) and report time (R-TIME) there is only one report starting date (R-SDATE) and report ending date (R-EDATE), that is, a one-to-one mapping;

(53) F-UNITNO , that is, the flying unit number by itself;

(54) A-NO <--<-------> A-TYPE, that is, for a given aircraft number (A-NO) there may be many aircraft type (A-TYPE); and

(55) A-NO,R-SDATE,R-EDATE <------------> S-ITNAME, T-PICOQY, S-ITUNIT, that is, for a given aircraft number (A-NO) with reporting starting date (R-SDATE) and report ending date (R-EDATE), there may be many sortie item name (S-ITNAME), periodic total items consumed quantities (T-PICOQY), and sortie items units of measure (S-ITUNIT).

c) The third normal form relations for the Consumed Item per Aircraft Report are:

(51) R-NO <------------> R-NAME;

(52) R-NO,R-DATE,R-TIME <------------> R-SDATE, R-EDATE;

(53) F-UNITNO

(54) A-NO <--<---------> A-TYPE; and

(55) A-NO,R-SDATE,R-EDATE <--<--------->

S-ITNAME, T-PICOQY, S-ITUNIT.

Report 11 - Consumed Item Quantity

a) The data items representing the entities of this report are: F-UNITNO, R-NO, R-NAME, R-DATE, R-TIME, R-SDATE, R-EDATE, S-ITNAME, S- ITSUPP, S-DEPLOC, S-DEPDAT, S-DEPTIM, A-TYPE, A-NO, S-ITCOQY, and S-ITRCNO.

b) The relationships between the data items of this report are:

(56) R-NO <----------> R-NAME, that means, for a given report number (R-NO) there is only one report name (R-NAME), that is, a one-to-one mapping represented as <-------> a two opposite arrows;

(57) R-NO,R-DATE,R-TIME <----------> R-SDATE, R-EDATE, that means, for a given report number (R-NO) with report date (R-DATE) and report time (R-TIME) there is only one report starting date (R-SDATE) and report ending date (R-EDATE), that is, a one-to-one mapping;

(58) F-UNITNO , that is, the flying unit number by itself;

(59) A-NO <--<-------> A-TYPE, that is, for a given aircraft number (A-NO) there may be many aircraft type (A-TYPE);

(60) S-ITRCNO <--<---------> S-ITNAME, S-ITCOQY,

S-ITSUPP, that is, for a given sortie item recept number (S-ITRCNO) there may be many sortie items names (S-ITNAME), sortie item consumed quantities (S-ITCOQY), and sortie items suppliers (S-ITSUPP); and

(61) A-NO,S-DEPDAT,S-DEPTIM <--<---------> S-ITRCNO, S-DEPLOC, that is, for a given aircraft number (A-NO), sortie departure date (S-DEPDAT), and sortie departure time (S-DEPTIM), there may be many sortie item receipts (S-ITRCNO) and sortie departure locations (S-DEPLOC).

c) The third normal form relations for the Consumed Item Quantity Report are:

(56) R-NO <------------> R-NAME;

(57) R-NO,R-DATE,R-TIME <------------> R-SDATE, R-EDATE;

(58) F-UNITNO

(59) A-NO <--<---------> A-TYPE;

(60) S-ITRCNO <--<---------> S-ITNAME, S-ITCOQY, S-ITSUPP; and

(61) A-NO,S-DEPDAT,S-DEPTIM <--<---------> S-ITRCNO, S-DEPLOC.

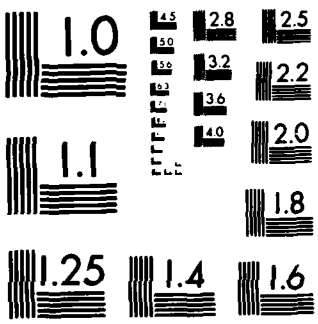
Report 12 - Crew Member's Summary

a) The data items representing the entities of this report

AD-A151 848 A DATABASE DESIGN FOR THE BRAZILIAN AIR FORCE FLYING UNIT OPERATIONAL CON..(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. A M DA CUNHA 3/3

UNCLASSIFIED 14 DEC 84 AFIT/GCS/ENG/84D-7 F/G 9/2 NL

END
FILMED
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963 A

are: F-UNITNO, R-NO, R-NAME, R-DATE, R-TIME, R-SDATE, R-EDATE, C-SSN, C-LNAME, C-CNAME, C-RANK, C-SPEC, C-FQCODE, and T-PCXFUN.

b) The relationships between the data items of this report are:

(62) R-NO <----------> R-NAME, that means, for a given report number (R-NO) there is only one report name (R-NAME), that is, a one-to-one mapping represented as <-------> a two opposite arrows;

(63) R-NO,R-DATE,R-TIME <----------> R-SDATE, R-EDATE, that means, for a given report number (R-NO) with report date (R-DATE) and report time (R-TIME) there is only one report starting date (R-SDATE) and report ending date (R-EDATE), that is, a one-to-one mapping;

(64) F-UNITNO <--<---------> C-SSN, that is, for a given flying unit number (F-UNITNO) there may be many crew members social security numbers (C-SSN);

(65) C-SSN <-----------> C-LNAME, C-CNAME, C-RANK, C-SPEC, C-FQCODE, that is, for a given crew member (C-SSN), there is only one crew last name (C-LNAME), crew complement name (C-CNAME), crew rank (C-RANK), crew specialty (C-SPECIALTY), and crew functional qualification code (C-FQCODE);

and

(66) C-SSN,R-SDATE,R-EDATE <------------> T-PCXFUN, that is, for a given crew member social security number (C-SSN) with report starting date (R-SDATE) and report ending date (R-EDATE), there only one periodic total of a crew member executed function (T-PCXFUN).

c) The third normal form relations for the Crew Member's Summary Report are:

(62) R-NO <------------> R-NAME;

(63) R-NO,R-DATE,R-TIME <------------> R-SDATE, R-EDATE;

(64) F-UNITNO <--<---------> C-SSN

(65) C-SSN <-----------> C-LNAME, C-CNAME, C-RANK, C-SPEC, C-FQCODE; and

(66) C-SSN,R-SDATE,R-EDATE <------------> T-PCXFUN.

Report 13 - Crew Member Individual Totals

a) The data items representing the entities of this report are: F-UNITNO, R-NO, R-NAME, R-DATE, R-TIME, R-SDATE, R-EDATE, C-SSN, C-RANK, C-SPEC, C-LNAME, C-CNAME, T-CF1TIM, T-CF1LND, T-CF2TIM, T-CF2LND, T-CF3TIM, T-CF3LND, T-PMSTIM, T-YCSTIM, G-TCSTIM, and G-TLNDNO.

b) The relationships between the data items of this report are:

(67) R-NO <----------> R-NAME, that means, for a

given report number (R-NO) there is only one report name (R-NAME), that is, a one-to-one mapping represented as <-------> a two opposite arrows;

(68) R-NO,R-DATE,R-TIME <----------> R-SDATE, R-EDATE, that means, for a given report number (R-NO) with report date (R-DATE) and report time (R-TIME) there is only one report starting date (R-SDATE) and report ending date (R-EDATE), that is, a one-to-one mapping;

(69) F-UNITNO <--<---------> C-SSN, that is, for a given flying unit number (F-UNITNO) there may be many crew members social security numbers (C-SSN);

(70) C-SSN <-----------> C-RANK, C-SPEC, C-LNAME, C-CNAME, T-CF1TIM, T-CF1LND, T-CF2TIM, T-CF2LND, T-CF3TIM, T-CF3LND, G-TLNDNO, G-TCSTIM, that is, for a given crew member (C-SSN) there is only one crew rank (C_RANK), crew specialty (C-SPEC), crew last name (C-LNAME), crew complement name (C-CNAME), total crew times executing sortie missions functions type 1 (T-CF1TIM), total crew landings executing sortie missions functions type 1 (T-CF1LND), total crew times executing sortie missions functions type 2 (T-CF2TIM), total crew landings executing sortie missions

functions type 2 (T-CF2LND), total crew times executing sortie missions functions type 3 (T-CF3TIM), total crew landings executing sortie missions functios type 3 (T-CF3LND), general total landing numbers per sorties (G-TLNDNO), and general total crew sortie time (G-TCSTIM);

(71) C-SSN,D-YEAR <--<---------> T-YCSTIM, that is, for a given crew member (C-SSN) and data year reference (D-YEAR), there may be several yearly total crew members sortie times (T-YCSTIM); and

(72) C-SSN,R-SDATE,R-EDATE <--<---------> T-PMSTIM, that is, for a given crew member social security number (C-SSN) with report starting date (R-SDATE) and report ending date (R-EDATE), there may be many periodic totals of mission sortie times (T-PMSTIM).

c) The third normal form relations for the Crew Member Individual Totals Report are:

(67) R-NO <------------> R-NAME;

(68) R-NO,R-DATE,R-TIME <------------> R-SDATE, R-EDATE;

(69) F-UNITNO <--<---------> C-SSN;

(70) C-SSN <-----------> C-RANK, C-SPEC, C-LNAME, C-CNAME, T-CF1TIM, T-CF1LND, T-CF2TIM, T-CF2LND, T-CF3TIM, T-CF3LND, G-TLNDNO, G-TCSTIM;

(71) C-SSN,D-YEAR <--<---------> T-YCSTIM; and

(72) C-SSN,R-SDATE,R-EDATE <--<---------> T-PMSTIM.

Report 14 - Crew Member's Totals Summary

a) The data items representing the entities of this report are: F-UNITNO, R-NO, R-NAME, R-DATE, R-TIME, R-SDATE, R-EDATE, C-SSN, C-LNAME, T-CF1TIM, T-CF1LND, T-CF2TIM, T-CF2LND, T-CF3TIM, T-CF3LND, T-PMSTIM, T-YCSTIM, G-TCSTIM, and G-TLNDNO.

b) The relationships between the data items of this report are:

(73) R-NO <-----------> R-NAME, that means, for a given report number (R-NO) there is only one report name (R-NAME), that is, a one-to-one mapping represented as <-------> a two opposite arrows;

(74) R-NO,R-DATE,R-TIME <-----------> R-SDATE, R-EDATE, that means, for a given report number (R-NO) with report date (R-DATE) and report time (R-TIME) there is only one report starting date (R-SDATE) and report ending date (R-EDATE), that is, a one-to-one mapping;

(75) F-UNITNO <--<---------> C-SSN, that is, for a given flying unit number (F-UNITNO) there may be many crew members social security numbers (C-SSN);

(76) C-SSN <--<---------> C-LNAME, T-CF1TIM,

T-CF1LND, T-CF2TIM, T-CF2LND, T-CF3TIM, T-CF3LND, G-TLNDNO, G-TCSTIM, that is, for a given crew member (C-SSN) there is only one crew last name (C-LNAME), total crew times executing sortie missions functions type 1 (T-CF1TIM), total crew landings executing sortie missions functions type 1 (T-CF1LND), total crew times executing sortie missions functions type 2 (T-CF2TIM), total crew landings executing sortie missions functions type 2 (T-CF2LND), total crew times executing sortie missions functions type 3 (T- CF3TIM), total crew landings executing sortie missions functios type 3 (T-CF3LND), general total landing numbers per sorties (G-TLNDNO), and general total crew sortie time (G-TCSTIM);

(77) C-SSN,D-YEAR <--<---------> T-YCSTIM, that is, for a given crew member (C-SSN) and data year reference (D-YEAR), there may be several yearly total crew members sortie times (T-YCSTIM); and

(78) C-SSN,R-SDATE,R-EDATE <--<---------> T-PMSTIM, that is, for a given crew member social security number (C-SSN) with report starting date (R-SDATE) and report ending date (R-EDATE), there may be many periodic totals of mission sortie times (T-PMSTIM).

c) The third normal form relations for the Crew Member's Totals Summary Report are:

(73) R-NO <------------> R-NAME;

(74) R-NO,R-DATE,R-TIME <------------> R-SDATE, R-EDATE;

(75) F-UNITNO <--<---------> C-SSN;

(76) C-SSN <--<---------> C-LNAME, T-CF1TIM, T-CF1LND, T-CF2TIM, T-CF2LND, T-CF3TIM, T-CF3LND, G-TLNDNO, G-TCSTIM;

(77) C-SSN,D-YEAR <--<---------> T-YCSTIM, that is ; and

(78) C-SSN,R-SDATE,R-EDATE <--<---------> T-PMSTIM.

Appendix D

The First Implementation Level

(Dialog Generator Management Software Prototype)

The idea of conceptualizing and implementing a Dialog Generator Management Software (DGMSW) Prototype before have designed a database system was born during the system analysis phase of the database design. Trying to understand the behavior of the entire system environment from the decision making process point of view, the author found out that whichever database need to be designed for a considered system environment, definitively it has to be part of some Decision Support System Architectural Model, integrated in some degree with a Statistical base and a model base, as shown in Figure 3 from Section 3.3 (26).

A program or group of programs were developed in order to demonstrate the application of the theoretical concepts of a Decision Support System directly to the system environment. The study and implementation of these menu-driven programs helped not only to analyze and understand the overall system environment, but also to demonstrate that the use of tools like a Dialog Generation Management Software can represent and explain by itself how different softwares could be integrated in just one unique management environment, grouping pieces of several different decision making processes. A very useful and necessary tool like that, if implemented in different system levels, activated from microcomputers, subjected to several levels of security could represent an optimum solution for complex Decision Support Systems (26).

In this thesis effort, only the conceptual part of a Decision Support System using menu-driven programs as a prototype of a Dialog Generator Management Software was addressed, projected, and implemented, helping to understand where designed databases are located to support decision making process. As summarized in Figure 19 from Section 5.1, the following sample of programs constitutes a prototype Dialog Generator Management Software (DGMSW) for the Brazilian Air Force Systems and related Systems Environment:

Figure 41 - The Environmental DSS Main Menu,
Figure 42 - The Environmental DSS Main Menu Program,
Figure 43 - National Defense DSS Main Menu,
Figure 44 - National Defense DSS Main Menu Program,
Figure 45 - Air Force DSS Menu,
Figure 46 - Air Force DSS Menu Program,
Figure 47 - Air Force Operations DSS Menu,
Figure 48 - Air Force Operations DSS Menu Program,
Figure 49 - Flying Unit Operations DSS Menu,
Figure 50 - Flying Unit Operations DSS Menu Program,
Figure 51 - FLUNITOC Specific DSS Menu,
Figure 52 - FLUNITOC Specific DSS Program,
Figure 53 - Model Base Management Software Menu,
Figure 54 - Model Base Management Software Menu Program,
Figure 55 - Statistical Analysis Management Software Menu,
Figure 56 - Statist. Analysis Mngmt Software Menu Program,
Figure 57 - Database Management Software Menu, and
Figure 58 - Database Management Software Menu Program.

A database management software (DBMSW) menu partial implementation and a Report Generation implementation are addressed in Appendix F using INGRES DBMS.

```
****** ENVIRONMENTAL DECISION SUPPORT SYSTEMS (DSS) ******

                        MAIN MENU

OPTIONS:

   1   NATIONAL DEFENSE DSS
   2   AIR FORCE DSS
   3   AIR FORCE OPERATIONS DSS
   3   FLYING UNIT OPERATIONS DSS
   4   FLYING UNIT OPERATIONAL CONTROL SPECIFIC DSS
   5   RESTARTING DATABASE
   6   EXIT TO OPERATING SYSTEM
   7   HELP MENU

SELECT OPTION:
```

Figure 41 - The Environmental DSS Main Menu

```
*main menu program for environmental decision support
systems
*by maj. cunha
*menu
*10/15/84
*initialize system
SET TALK OFF
SELECT PRIMARY
*SET FORMAT TO SCREEN
SET FORMAT TO PRINT
SET PRINT ON
SET CONSOLE ON
*scan for input option
```

```
DO WHILE t
        CLEAR
        STORE ' DECISION SUPPORT SYSTEMS (DSS) ' to mtitle
        SAVE TO B:title
        ERASE
        @1,1 SAY '             ****** '  +mtitle+ '******'
        @2,1 SAY 'MAIN MENU'
        @2,60 SAY DATE( )
        @4,1 SAY 'OPTIONS:'
        @5,1 SAY '    1    NATIONAL DEFENSE DSS'
        @6,1 SAY '    2    AIR FORCE DSS'
        @7,1 SAY '    3    AIR FORCE OPERATIONS DSS'
        @8,1 SAY '    4    FLYING UNIT OPERATIONS DSS'
        @9,1 SAY '    5    FLYING UNIT OPERATIONAL CONTROL
SPECIFIC DSS'
        @10,1 SAY '    6    RESTARTING DATABASE'
        @11,1 SAY '    7    EXIT TO OPERATING SYSTEM'
        @12,1 SAY '    8    HELP MENU'
        @19,1 SAY 'SELECT OPTION :'
        STORE ' ' TO option
        WAIT TO option
*  Check for valid input and branch to appropriate submenu
        do CASE
             CASE option = '1'
                DO b:nddss.cmd
             CASE option = '2'
                DO b:afdss.cmd
             CASE option = '3'
                DO b:afodss.cmd
             CASE option = '4'
                DO b:fuodss.cmd
             CASE option = '5'
                DO b:fuocsdss.cmd
             CASE option = '6'
                CANCEL
             CASE option = '7'
                CLEAR
                QUIT
             CASE option = '8'
                DO b:helpmenu.cmd
        OTHERWISE
             @ 23,1 SAY  'ILLEGAL OPTION'
             STORE 1 TO xx
             Do WHILE xx < 35
                STORE xx + 1 to xx
             ENDDO
        ENDCASE
ENDDO WHILE t
```

Figure 42 - The Environmental DSS Menu Program

```
**********  DECISION SUPPORT SYSTEMS (DSS)  **********

***************  NATIONAL DEFENSE DSS  ***************

MENU:

OPTIONS:

    1   AIR FORCE DSS
    2   ARMY DSS
    3   NAVY DSS
    ...

    4   ECONOMICS DSS
    5   RETURN TO PREVIOUS MENU
    6   EXIT TO OPERATING SYSTEM
    7   HELP

SELECT OPTION:
```

Figure 43 - National Defense DSS Menu

```
*main menu for decision support system
*by maj. cunha
*menu
*10/15/84
*initialize system
SET TALK OFF
SELECT PRIMARY
*SET FORMAT TO SCREEN
SET FORMAT TO PRINT
```

```
SET PRINT ON
SET CONSOLE ON
*scan for input option
DO WHILE t
       CLEAR
       STORE ' DECISION SUPPORT SYSTEMS (DSS) ' to mtitle
       SAVE TO B:title
       ERASE
       @1,1 SAY '            ****** '  +mtitle+ '******'
       @2,1 SAY 'MENU'
       @2,60 SAY DATE( )
       @4,1 SAY 'OPTIONS:'
       @5,1 SAY '    1    AIR FORCE DSS'
       @6,1 SAY '    2    ARMY DSS'
       @7,1 SAY '    3    NAVY DSS'
       @8,1 SAY '    4    ECONOMICS DSS'
       @9,1 SAY '    5    RETURN TO PREVIOUS MENU'
       @10,1 SAY '    6    EXIT TO OPERATING SYSTEM'
       @11,1 SAY '    7    HELP MENU'
       @19,1 SAY 'SELECT OPTION :'
       STORE ' ' TO option
       WAIT TO option
*  Check for valid input and branch to appropriate submenu
       do CASE
            CASE option = '1'
               DO b:afdss.cmd
            CASE option = '2'
               DO b:armydss.cmd
            CASE option = '3'
               DO b:navydss.cmd
            CASE option = '4'
               DO b:econodss.cmd
            CASE option = '5'
               CANCEL
            CASE option = '6'
               CLEAR
               QUIT
            CASE option = '7'
               DO b:helpmenu.cmd
       OTHERWISE
            @ 23,1 SAY  'ILLEGAL OPTION'
            STORE 1 TO xx
            Do WHILE xx < 35
               STORE xx + 1 to xx
            ENDDO
       ENDCASE
ENDDO WHILE t
```

Figure 44 - National Defense DSS Menu Program

```
**********  DECISION SUPPORT SYSTEMS (DSS)  **********

******************  AIR FORCE DSS  ******************

MENU:

OPTIONS:

   1   AIR FORCE PERSONNEL DSS
   2   AIR FORCE LOGISTICS DSS
   3   AIR FORCE TRAINING DSS
   ...
   4   AIR FORCE OPERATIONS DSS
   5   RETURN TO PREVIOUS MENU
   6   EXIT TO OPERATING SYSTEM
   7   HELP

SELECT OPTION:
```

Figure 45 - Air Force DSS Menu

```
*main menu for decision support system
*by maj. cunha
*menu
*10/15/84
*initialize system
SET TALK OFF
SELECT PRIMARY
*SET FORMAT TO SCREEN
```

```
SET FORMAT TO PRINT
SET PRINT ON
SET CONSOLE ON
*scan for input option
DO WHILE t
        CLEAR
        STORE ' DECISION SUPPORT SYSTEMS (DSS) ' to mtitle
        SAVE TO B:title
        ERASE
        @1,1 SAY '             ****** '  +mtitle+ '******'
        @2,1 SAY 'MENU'
        @2,60 SAY DATE( )
        @4,1 SAY 'OPTIONS:'
        @5,1 SAY '    1    AIR FORCE PERSONNEL DSS'
        @6,1 SAY '    2    AIR FORCE LOGISTICS DSS'
        @7,1 SAY '    3    AIR FORCE TRAINING DSS'
        @8,1 SAY '    4    AIR FORCE OPERATIONS DSS'
        @9,1 SAY '    5    RETURN TO PREVIOUS MENU'
        @10,1 SAY '    6    EXIT TO OPERATING SYSTEM'
        @11,1 SAY '    7    HELP MENU'
        @19,1 SAY 'SELECT OPTION :'
        STORE ' ' TO option
        WAIT TO option
*  Check for valid input and branch to appropriate submenu
        do CASE
             CASE option = '1'
                DO b:afpdss.cmd
             CASE option = '2'
                DO b:afldss.cmd
             CASE option = '3'
                DO b:aftdss.cmd
             CASE option = '4'
                DO b:afodss.cmd
             CASE option = '5'
                CANCEL
             CASE option = '6'
                CLEAR
                QUIT
             CASE option = '7'
                DO b:helpmenu.cmd
        OTHERWISE
             @ 23,1 SAY  'ILLEGAL OPTION'
             STORE 1 TO xx
             Do WHILE xx < 35
                STORE xx + 1 to xx
             ENDDO
        ENDCASE
ENDDO WHILE t
```

Figure 46 - Air Force DSS Menu Program

```
**********  DECISION SUPPORT SYSTEMS (DSS)  **********

*************  AIR FORCE OPERATIONS DSS  *************

MENU:

OPTIONS:

   1   FLYING UNIT OPERATIONS DSS
   2   ADMINISTRATIVE UNIT OPERATIONS DSS
   3   TRAINING UNIT OPERATIONS DSS
   ...
   4   LOGISTICS UNIT OPERATIONS DSS
   5   RETURN TO PREVIOUS MENU
   6   EXIT TO OPERATING SYSTEM
   7   HELP

SELECT OPTION:
```

Figure 47 - Air Force Operations DSS Menu

```
*main menu for decision support system
*by maj. cunha
*menu
*10/15/84
*initialize system
SET TALK OFF
SELECT PRIMARY
*SET FORMAT TO SCREEN
SET FORMAT TO PRINT
```

```
SET PRINT ON
SET CONSOLE ON
*scan for input option
DO WHILE t
        CLEAR
        STORE ' DECISION SUPPORT SYSTEMS (DSS) ' to mtitle
        SAVE TO B:title
        ERASE
        @1,1 SAY '             ****** '  +mtitle+ '******'
        @2,1 SAY 'MENU'
        @2,60 SAY DATE( )
        @4,1 SAY 'OPTIONS:'
        @5,1 SAY '    1    FLYING UNIT OPERATIONS DSS'
        @6,1 SAY '    2    ADMINISTRATIVE UNIT OPERATIONS
DSS'
        @7,1 SAY '    3    TRAINING UNIT OPERATIONS DSS'
        @8,1 SAY '    4    LOGISTICS UNIT OPERATIONS DSS'
        @9,1 SAY '    5    RETURN TO PREVIOUS MENU'
        @10,1 SAY '    6    EXIT TO OPERATING SYSTEM'
        @11,1 SAY '    7    HELP MENU'
        @19,1 SAY 'SELECT OPTION :'
        STORE ' ' TO option
        WAIT TO option
*  Check for valid input and branch to appropriate submenu
        do CASE
             CASE option = '1'
                DO b:fuodss.cmd
             CASE option = '2'
                DO b:auodss.cmd
             CASE option = '3'
                DO b:tuodss.cmd
             CASE option = '4'
                DO b:luodss.cmd
             CASE option = '5'
                CANCEL
             CASE option = '6'
                CLEAR
                QUIT
             CASE option = '7'
                DO b:helpmenu.cmd
        OTHERWISE
             @ 23,1 SAY  'ILLEGAL OPTION'
             STORE 1 TO xx
             Do WHILE xx < 35
                STORE xx + 1 to xx
             ENDDO
        ENDCASE
ENDDO WHILE t
```

Figure 48 - Air Force Operations DSS Menu Program

```
   **********  DECISION SUPPORT SYSTEMS (DSS)  **********

   ************  FLYING UNIT OPERATIONS DSS  ************

MENU:


OPTIONS:

   1   FLYING UNIT OPERATIONAL PLANNING SPECIFIC DSS
   2   FLYING UNIT OPERATIONAL CONTROL SPECIFIC DSS
   3   FLYING UNIT TRAINING CONTROL SPECIFIC DSS
   ...

   4   FLYING UNIT FLIGHT SAFETY SPECIFIC DSS
   5   RETURN TO PREVIOUS MENU
   6   EXIT TO OPERATING SYSTEM
   7   HELP


SELECT OPTION:
```

Figure 49 - Flying Unit Operations DSS Menu

```
*main menu for decision support system
*by maj. cunha
*menu
*10/15/84
*initialize system
SET TALK OFF
SELECT PRIMARY
*SET FORMAT TO SCREEN
SET FORMAT TO PRINT
SET PRINT ON
SET CONSOLE ON
```

```
*scan for input option
DO WHILE t
        CLEAR
        STORE ' DECISION SUPPORT SYSTEMS (DSS) ' to mtitle
        SAVE TO B:title
        ERASE
        @1,1 SAY '             ****** '  +mtitle+ '******'
        @2,1 SAY 'MENU'
        @2,60 SAY DATE( )
        @4,1 SAY 'OPTIONS:'
        @5,1 SAY '    1    FLYING UNIT OPERATIONAL PLANNING
SPECIFIC DSS'
        @6,1 SAY '    2    FLYING UNIT OPERATIONAL CONTROL
SPECIFIC DSS'
        @7,1 SAY '    3    FLYING UNIT TRAINING CONTROL
SPECIFIC DSS'
        @8,1 SAY '    4    FLYING UNIT FLIGHT SAFETY
SPECIFIC DSS'
        @9,1 SAY '    5    RETURN TO PREVIOUS MENU'
        @10,1 SAY '    6    EXIT TO OPERATING SYSTEM'
        @11,1 SAY '    7    HELP MENU'
        @19,1 SAY 'SELECT OPTION :'
        STORE ' ' TO option
        WAIT TO option
*  Check for valid input and branch to appropriate submenu
        do CASE
             CASE option = '1'
                DO b:fuopsdss.cmd
             CASE option = '2'
                DO b:fuocsdss.cmd
             CASE option = '3'
                DO b:futcdss.cmd
             CASE option = '4'
                DO b:fufssdss.cmd
             CASE option = '5'
                CANCEL
             CASE option = '6'
                CLEAR
                QUIT
             CASE option = '7'
                DO b:helpmenu.cmd
        OTHERWISE
             @ 23,1 SAY  'ILLEGAL OPTION'
             STORE 1 TO xx
             Do WHILE xx < 35
                STORE xx + 1 to xx
             ENDDO
        ENDCASE
 ENDDO WHILE t
```

Figure 50 - Flying Unit Operations DSS Menu Program

```
********** DECISION SUPPORT SYSTEMS (DSS) **********

*** FLYING UNIT OPERATIONAL CONTROL SPECIFIC DSS ***

MENU:

OPTIONS:

   1   MODEL BASE MANAGEMENT SOFTWARE (MBMSW)
   2   STATISTICS ANALYSIS MANAGEMENT SOFTWARE (SAMSW)
   3   DATA BASE MANAGEMENT SOFTWARE (DBMSW)
   4   RETURN TO PREVIOUS MENU
   5   EXIT TO OPERATING SYSTEM
   6   HELP

SELECT OPTION:
```

Figure 51 Flying Unit Operational Control Specific DSS Menu

```
*main menu for decision support system
*by maj. cunha
*menu
*10/15/84
*initialize system
SET TALK OFF
SELECT PRIMARY
*SET FORMAT TO SCREEN
SET FORMAT TO PRINT
```

```
SET PRINT ON
SET CONSOLE ON
*scan for input option
DO WHILE t
        CLEAR
        STORE ' DECISION SUPPORT SYSTEMS (DSS) ' to mtitle
        SAVE TO B:title
        ERASE
        @1,1 SAY '              ****** '  +mtitle+ '******'
        @2,1 SAY 'MENU'
        @2,60 SAY DATE( )
        @4,1 SAY 'OPTIONS:'
        @5,1 SAY '    1    MODEL BASE MANAGEMENT SOFTWARE
(MBMSW)'
        @6,1 SAY '    2    STATISTICAL ANALYSIS MANAGEMENT
SOFTWARE (SAMSW)'
        @7,1 SAY '    3    DATA BASE MANAGEMENT SOFTWARE
(DBMSW)'
        @8,1 SAY '    4    RETURN TO PREVIOUS MENU'
        @9,1 SAY '    5    EXIT TO OPERATING SYSTEM'
        @10,1 SAY '    6    HELP MENU'
        @19,1 SAY 'SELECT OPTION :'
        STORE ' ' TO option
        WAIT TO option
*  Check for valid input and branch to appropriate submenu
        do CASE
             CASE option = '1'
                DO b:mbmsw.cmd
             CASE option = '2'
                DO b:samsw.cmd
             CASE option = '3'
                DO b:dbmsw.cmd
             CASE option = '4'
                CANCEL
             CASE option = '5'
                CLEAR
                QUIT
             CASE option = '6'
                DO b:helpmenu.cmd
        OTHERWISE
             @ 23,1 SAY  'ILLEGAL OPTION'
             STORE 1 TO xx
             Do WHILE xx < 35
                STORE xx + 1 to xx
             ENDDO
        ENDCASE
ENDDO WHILE t
```

Figure 52 Flying Unit Operational Control SDSS Menu Program

```
    ***  FLYING UNIT OPERATIONAL CONTROL SPECIFIC DSS  ***

    ******  MODEL BASE MANAGEMENT SOFTWARE (MBMSW)  ******

MENU:

OPTIONS:

   1   PASCAL
   2   FORTRAN
   3   SLAM
   ...

   4   EQUATIONS
   5   RETURN TO PREVIOUS MENU
   6   EXIT TO OPERATING SYSTEM
   7   HELP

SELECT OPTION:
```

Figure 53 Model Base Management Software (MBMSW) Menu

```
*main menu for decision support system
*by maj. cunha
*menu
*10/15/84
*initialize system
SET TALK OFF
SELECT PRIMARY
*SET FORMAT TO SCREEN
```

```
SET FORMAT TO PRINT
SET PRINT ON
SET CONSOLE ON
*scan for input option
DO WHILE t
        CLEAR
        STORE ' FLYING UNIT OPERATIONAL CONTROL SPECIFIC DSS
' to mtitle
        SAVE TO B:title
        ERASE
        @1,1 SAY '             *** '  +mtitle+ '***'
        @2,1 SAY 'MENU'
        @2,60 SAY DATE( )
        @4,1 SAY 'OPTIONS:'
        @5,1 SAY '    1    PASCAL'
        @6,1 SAY '    2    FORTRAN'
        @7,1 SAY '    3    SLAM'
        @8,1 SAY '    4    EQUATIONS'
        @9,1 SAY '    5    RETURN TO PREVIOUS MENU'
        @10,1 SAY '    6    EXIT TO OPERATING SYSTEM'
        @11,1 SAY '    7    HELP MENU'
        @19,1 SAY 'SELECT OPTION :'
        STORE ' ' TO option
        WAIT TO option
*  Check for valid input and branch to appropriate submenu
        do CASE
             CASE option = '1'
                DO b:pascal.cmd
             CASE option = '2'
                DO b:fortran.cmd
             CASE option = '3'
                DO b:slam.cmd
             CASE option = '4'
                DO b:equation.cmd
             CASE option = '5'
                CANCEL
             CASE option = '6'
                CLEAR
                QUIT
             CASE option = '7'
                DO b:helpmenu.cmd
        OTHERWISE
             @ 23,1 SAY  'ILLEGAL OPTION'
             STORE 1 TO xx
             Do WHILE xx < 35
                STORE xx + 1 to xx
             ENDDO
        ENDCASE
 ENDDO WHILE t
```

Figure 54 Model Base Management Software Menu Program

```
***  FLYING UNIT OPERATIONAL CONTROL SPECIFIC DSS  ***

*  STATISTICAL ANALYSIS MANAGEMENT SOFTWARE (SAMSW)   *

MENU:


OPTIONS:

    1   S PACKAGE
    2   SPSS PACKAGE
    3   GRAPHS
    4   PLOTS
    ...

    5   MISCELANEOUS
    6   RETURN TO PREVIOUS MENU
    7   EXIT TO OPERATING SYSTEM
    8   HELP


SELECT OPTION:
```

Figure 55 Statistical Analysis Management Software Menu

```
*main menu for decision support system
*by maj. cunha
*menu
*10/15/84
*initialize system
SET TALK OFF
SELECT PRIMARY
*SET FORMAT TO SCREEN
SET FORMAT TO PRINT
SET PRINT ON
SET CONSOLE ON
```

```
*scan for input option
DO WHILE t
        CLEAR
        STORE ' FLYING UNIT OPERATIONAL CONTROL SPECIFIC DSS
' to mtitle
        SAVE TO B:title
        ERASE
        @1,1 SAY '              ****** '  +mtitle+ '******'
        @2,1 SAY 'MENU'
        @2,60 SAY DATE( )
        @4,1 SAY 'OPTIONS:'
        @5,1 SAY '    1    S PACKAGE'
        @6,1 SAY '    2    SPSS PACKAGE'
        @7,1 SAY '    3    GRAPHS'
        @8,1 SAY '    4    PLOTS'
        @9,1 SAY '    5    MISCELANEOUS'
        @10,1 SAY '    6    RETURN TO PREVIOUS MENU'
        @11,1 SAY '    7    EXIT TO OPERATING SYSTEM'
        @12,1 SAY '    8    HELP MENU'
        @19,1 SAY 'SELECT OPTION :'
        STORE ' ' TO option
        WAIT TO option
*  Check for valid input and branch to appropriate submenu
        do CASE
             CASE option = '1'
                DO b:s.cmd
             CASE option = '2'
                DO b:spss.cmd
             CASE option = '3'
                DO b:graphs.cmd
             CASE option = '4'
                DO b:plots.cmd
             CASE option = '5'
                DO b:miscelane.cmd
             CASE option = '6'
                CANCEL
             CASE option = '7'
                CLEAR
                QUIT
             CASE option = '8'
                DO b:helpmenu.cmd
        OTHERWISE
             @ 23,1 SAY  'ILLEGAL OPTION'
             STORE 1 TO xx
             Do WHILE xx < 35
                STORE xx + 1 to xx
             ENDDO
        ENDCASE
ENDDO WHILE t
```

Figure 56 Statistical Analysis Management Software Menu Program

```
***  FLYING UNIT OPERATIONAL CONTROL SPECIFIC DSS  ***

******  DATA BASE MANAGEMENT SOFTWARE (DBMSW)  *******

MENU:

OPTIONS:

   1   TOTAL
   2   INGRES
   3   DBASE II
   4   RBASE
   5   RETURN TO PREVIOUS MENU
   6   EXIT TO OPERATING SYSTEM
   7   HELP

SELECT OPTION:
```

Figure 57 - Data Base Management Software (DBMSW) Menu

```
*main menu for decision support system
*by maj. cunha
*menu
*10/15/84
*initialize system
SET TALK OFF
SELECT PRIMARY
*SET FORMAT TO SCREEN
SET FORMAT TO PRINT
SET PRINT ON
SET CONSOLE ON
*scan for input option
DO WHILE t
```

```
        CLEAR
        STORE ' FLYING UNIT OPERATIONAL CONTROL SPECIFIC DSS
' to mtitle
        SAVE TO B:title
        ERASE
        @1,1 SAY '             ****** '  +mtitle+ '******'
        @2,1 SAY 'MENU'
        @2,60 SAY DATE( )
        @4,1 SAY 'OPTIONS:'
        @5,1 SAY '    1    TOTAL'
        @6,1 SAY '    2    INGRES'
        @7,1 SAY '    3    dBASE II'
        @8,1 SAY '    4    RBASE'
        @9,1 SAY '    5    RETURN TO PREVIOUS MENU'
        @10,1 SAY '    6    EXIT TO OPERATING SYSTEM'
        @11,1 SAY '    7    HELP MENU'
        @19,1 SAY 'SELECT OPTION :'
        STORE ' ' TO option
        WAIT TO option
*  Check for valid input and branch to appropriate submenu
        do CASE
             CASE option = '1'
                DO b:total.cmd
             CASE option = '2'
                DO b:ingres.cmd
             CASE option = '3'
                DO b:dbaseii.cmd
             CASE option = '4'
                DO b:rbase.cmd
             CASE option = '5'
                CANCEL
             CASE option = '6'
                CLEAR
                QUIT
             CASE option = '7'
                DO b:helpmenu.cmd
        OTHERWISE
             @ 23,1 SAY  'ILLEGAL OPTION'
             STORE 1 TO xx
             Do WHILE xx < 35
                STORE xx + 1 to xx
             ENDDO
        ENDCASE
ENDDO WHILE t
```

Figure 58 Data Base Management Software Menu Program

Appendix E

A Sample of Implemented Relations Using INGRES DBMS

Using the INGRES DBMS Version 7.10, the relational database management system installed in the AFIT VAX 11/780 computer, the following list of relations represent the sample adopted in order to support the second and third level of implementation:

1) reportid;
2) aircraft;
3) report;
4) mistype;
5) crew;
6) acftaval;
7) flunit; and
8) flyunit.

Each relation is presented in two parts: (a) implementing relations using INGRES data definition language (DDL), and (b) loading relation using sample data items. In part (a) each relation is created, saved until an extended time, modified in its access method whenever necessary (Heap is the INGRES default), and finally printed for demonstration. In part (b) each relation sample data item is created in separeted files, loaded into INGRES DBMS, and finally printed for demonstration.

1) The "reportid" relation.

(a) Implementing "reportid" relation.

```
Relation:                 reportid
Owner:                    acunha
Tuple width:              44
Saved until:              Thu Jan 31 00:00:00 1985
Number of tuples:         4
Storage structure:        random hash
Relation type:            user relation

 attribute name    type  length  Keyno.

 r_no               i       4       1
 r_name             c      40
```

(b) Loading "reportid" sample relation.

reportid relation

r_no	r_name
1	Individual Flight Record
2	Mission Type Summary
4	Aircraft Numbers Missions Summary
7	Items per Mission

2) The "aircraft" relation.

(a) Implementing "aircraft" relation.

```
Relation:                 aircraft
Owner:                    acunha
Tuple width:              14
Saved until:              Thu Jan 31 00:00:00 1985
Number of tuples:         6
Storage structure:        ISAM file
Relation type:            user relation

 attribute name    type  length  Keyno.

 a_no               i       4      1
 a_type             c      10
```

(b) Loading "aircraft" sample relation.

aircraft relation

a_no	a_type
2120	mirage
2121	mirage
2122	mirage
2220	f-16
2223	f-16
2301	f-15

3) The "report" relation.

(a) Implementing "report" relation.

```
Relation:                report
Owner:                   acunha
Tuple width:             20
Saved until:             Thu Jan 31 00:00:00 1985
Number of tuples:        24
Storage structure:       paged heap
Relation type:           user relation
```

attribute name	type	length	Keyno.
r_no	i	4	
r_date	i	4	
r_time	i	4	
r_sdate	i	4	
r_edate	i	4	

(b) Loading "report" sample relation.

report relation

r_no	r_date	r_time	r_sdate	r_edate
1	841216	100000	841201	841215
2	841216	100500	841201	841215
4	841216	101000	841201	841215
7	850116	0	850101	850115
7	2147479348	16700	841101	841231
7	2147479348	16700	841201	841231
7	2147479348	16707	841215	850101
7	2147479348	16707	840530	851010
7	841004	2059	841201	850130
7	841004	2124	841201	850130
7	841004	2127	841201	850201
7	841004	2232	841201	850301
7	841004	2234	841115	850401
7	841022	1	841201	850301
7	841022	7	841130	850130
7	841023	1649	841015	850215
7	841023	1651	841120	850130
7	841023	1652	840930	850331
7	841023	1653	841030	850215
7	841023	1654	841015	850410
7	841026	1335	841030	850215
7	841028	1329	841101	850215
7	841028	1534	841101	850331
7	841104	1636	841101	850315

4) The "mistype relation.
(a) Implementing "mistype" relation.

Relation:	mistype
Owner:	acunha
Tuple width:	58
Saved until:	Thu Jan 31 00:00:00 1985
Number of tuples:	11
Storage structure:	ISAM file
Relation type:	user relation

attribute name	type	length	Keyno.
a_no	i	4	1
s_mtcode	c	10	2
r_sdate	i	4	3
r_edate	i	4	4
s_time	f	4	
s_itcode	c	12	
s_itcoqy	i	4	
s_deploc	c	8	
s_arrloc	c	8	

(b) Loading "mistype" sample relation.

mistype relation

a_no	s_mtcode	r_sdate	r_edate
2121	intercept	841201	841215
2121	intercept	841201	841215
2120	combat	841201	841215
2220	grndsupp	841201	841215
2223	formation	841201	841215
2301	formation	841201	841215
2120	grndsupp	841217	850115
2120	combat	841219	850115
2223	combat	841219	850115
2223	intercept	841219	850115
2121	intercept	841219	850115

s_time	s_itcode	s_itcoqy	s_deploc	s_arrloc
2.700	miss77	2	sbbr	sbbr
1.300	miss24	1	sbbr	sbrj
0.700	gunshot	23	sbbr	sbbr
2.100	gunshot	36	sbbr	sbbr
1.400	rocket1	2	sbbr	sbbr
1.500	rocket3	4	sbbr	sbbr
6.200	bombmk76	6	sbbr	sbsc
2.300	bombmk102	2	sbsc	sbsc
1.200	gunshot	30	sbpa	sbpa
1.200	miss24	4	sbbr	sbpa
2.300	rocket5	2	sbbr	sbsc

5) The "crew" relation.

(a) Implementing "crew" relation.

Relation:	crew
Owner:	acunha
Tuple width:	81
Saved until:	Thu Jan 31 00:00:00 1985
Number of tuples:	5
Storage structure:	ISAM file
Relation type:	user relation

attribute name	type	length	Keyno.
c_ssn	i	4	1
c_rank	c	8	
c_spec	c	18	
c_lname	c	15	
c_cname	c	30	
c_fqcode	c	6	

(b) Loading "crew" sample relation.

crew relation

c_ssn	c_rank	c_spec
391801970	capt	pilot
291801970	major	pilot
691801613	ltcol	navigator
591811724	capt	flight engineer
491801923	capt	pilot

c_lname	c_cname	c_fqco
john		1p
cunha		ip
foster		1p
david		1f
ali		ip

6) The "acftaval" relation.

(a) Implementing "acftaval" relation.

Relation:	acftaval
Owner:	acunha
Tuple width:	26
Saved until:	Thu Jan 31 00:00:00 1985
Number of tuples:	5
Storage structure:	ISAM file
Relation type:	user relation

attribute name	type	length	keyno.
a_no	i	4	1
a_avdate	i	4	2
a_avtime	i	4	3
a_avperd	i	2	
a_avsitu	c	12	

(b) Loading "acftaval" sample relation.

acftaval relation

a_no	a_avdate	a_avtime	a_avpe	a_avsitu
2120	850114	100000	2	available
2122	850107	70000	1	situationc
2223	850114	100000	2	available
2301	850103	63000	1	available
2121	850112	210000	3	situationa

7) The "flunit" relation.

(a) Implementing "flunit" relation.

Relation:	flunit
Owner:	acunha
Tuple width:	16
Saved until:	Thu Jan 31 00:00:00 1985
Number of tuples:	6
Storage structure:	ISAM file
Relation type:	user relation

attribute name	type	length	Keyno.
f_unitno	i	4	1
c_ssn	i	4	
a_no	i	4	
f_unitrg	c	4	

(b) Loading "flunit" sample relation.

flunit relation

f_unitno	c_ssn	a_no	f_unit
2132732	291801970	2120	a
2132732	291801970	2220	a
2132732	391801970	2121	b
2132732	491801923	2122	c
3110234	591811724	2223	e
3110234	691801613	2301	d

8) The "flyunit" relation.

(a) Implementing "flyunit" relation.

```
Relation:                 flyunit
Owner:                    acunha
Tuple width:              12
Saved until:              Thu Jan 31 00:00:00 1985
Number of tuples:         6
Storage structure:        ISAM file
Relation type:            user relation
```

attribute name	type	length	Keyno.
f_unitno	i	4	1
c_ssn	i	4	
a_no	i	4	

(b) Loading "flyunit" sample relation.

flyunit relation

f_unitno	c_ssn	a_no
2132732	291801970	2120
2132732	291801970	2220
2132732	391801970	2121
2132732	491801923	2122
3110234	591811724	2223
3110234	691801613	2301

Appendix F

The Second Implementation Level

(The FLUNITOC Main Menu and Report Generator)

"QUEL" is the INGRES QUEry Language and "EQUEL" the Embedded QUEry Language specified in references (15), (16), and (34). They describe the comands and features which are used inside INGRES. The Embedded QUEL (EQUEL) provides the INGRES users with a method of interfacing the general purpose programming language "C".

In this Appendix, using EQUEL, a Flying Unit Operational Control Database System Main Menu is partially implemented and a Report Generator sample utilizing Report #7 is implemented.

Representing the second level of implementation the FLUNITOC Main Menu and the Report Generator are presented as follows through a documented application program.

```
/************************************************************
*                                                           *
*     DATE: 26 Nov 84                                       *
*     VERSION: 1.0                                          *
*                                                           *
*     TITLE: Main Menu and Report Generator Application     *
*            Program developed as second implementation     *
*            level for the AFIT thesis GCS/ENG/84D-7.       *
*     FILENAME: rg.c                                        *
*     OWNER: Maj. Adilson Marques da Cunha                  *
*            (BRAZILIAN AIR FORCE - BAF)                    *
*                                                           *
*     SOFTWARE SYSTEM: The Brazilian Air Force Flying       *
*                      Unit Operational Control (BAF        *
*                      FLUNITOC) Database System            *
*     OPERATING SYSTEM: UNIX                                *
*     LANGUAGE: INGRES EQUEL (Embedded query language       *
*               interface to "C")                           *
*     USE: database management software (DBMSW)            *
*     CONTENTS: FLUNITOC main menu & report generator #7    *
*               with the following modules:                 *
*               1.0 main menu,                              *
*               1.1 report_generator,                       *
*               1.1.1 build_head,                           *
*               1.1.1.1 print_head,                         *
*               1.1.2 build_consumption_report              *
*               1.1.2.1 sum_sortie_times                    *
*               1.1.2.1.1 find_match                        *
*               1.1.2.2 sum_items                           *
*               1.1.3 check_range_of_dates,                 *
*               1.1.3.1 check_date,                         *
*               1.1.4 get_day_time                          *
*                                                           *
*     FUNCTION: This program partially implements the BAF  *
*               FLUNITOC database system main menu and      *
*               implements the report generator option      *
*               using the report #7.                        *
*                                                           *
************************************************************/

#define NULLC '\0'
#define ERROR -1
#define MAX_MIS_TYPE 30/* max number of different mission */
                       /* types, i.e., interception, etc. */
#define MAX_ITEM_TYPE 30 /* max number of consumed items, */
                         /* i.e., rocket1, gunshot, etc.  */
#define MIS_LEN 11
#define LOC_LEN 9
#define ITEM_LEN 13
#define ACFT_LEN 11
#define TRUE 1
#define FALSE 0
```

```
/**************************************************************
*                                                             *
*    DATE: 26 Nov 84                                          *
*    VERSION: 1.0                                             *
*                                                             *
*    NAME: main menu                                          *
*    MODULE NUMBER: 1.0                                       *
*    FUNCTION: Prints FLUNITOC main menu to the screen.       *
*    INPUTS: None                                             *
*    OUTPUTS: None                                            *
*    GLOBAL VARIABLES USED: None                              *
*    GLOBAL VARIABLES CHANGED: None                           *
*    GLOBAL TABLES USED: None                                 *
*    GLOBAL TABLES CHANGED: None                              *
*    FILES READED: None                                       *
*    FILES WRITTEN: None                                      *
*    TERMINAL INPUT: options -- user menu choice              *
*    MODULES CALLED: report_generator                         *
*    CALLING MODULES: None                                    *
*                                                             *
*    AUTHOR: Maj. Adilson Marques da Cunha                    *
*            ( BRAZILIAN AIR FORCE)                           *
*    HISTORY: None                                            *
*                                                             *
**************************************************************/

main ()
  {

  int option;
  int quit;

##      ingres flunitoc
        quit = FALSE;
        while ( !quit )
        {
            printf ("\f\n\n\nBRAZILIAN AIR FORCE");
            printf ("\n\nFLYING UNIT OPERATIONAL CONTROL
SYSTEM");
            printf ("\n\n  ***  MAIN MENU  ***");
            printf
("\n\n-----------------------------------------
---------------------");
            printf ("\n\n\tOPTIONS:");
            printf ("\n\t    1\t\tData Entry");
            printf ("\n\t    2\t\tReport Generation");
            printf ("\n\t    3\t\tUpdate Transaction");
            printf ("\n\t    4\t\tQuery Transaction");
            printf ("\n\t    5\t\tHelp");
            printf ("\n\t    6\t\tQuit\n\n-->  ");
```

```
                    scanf ("%d", &option);
                    switch (option)   /* go to user option */

                    {
          case 2:
                        report_generator ();
                        break;
          case 6:       quit = TRUE;
                        break;
          case 1:
          case 3:
          case 4:
          case 5:       printf ("\nOption %d not yet implemented.",
                        option);
                        break;
          default:
                        printf ("\nInvalid options %d, please try
          again.",
                        option);
                    }
                }
          ##      exit
            }
```

```
/*************************************************************
*    DATE: 26 Nov 84                                         *
*    VERSION: 1.0                                            *
*    NAME: report_generator                                  *
*    MODULE NUMBER: 1.1                                      *
*    FUNCTION: This module gets the users' inputs in order *
*              to generate the appropriate report            *
*    TERMINAL INPUTS: report_num, fly_id, date               *
*    OUTPUTS: None                                           *
*    GLOBAL VARIABLES USED: None                             *
*    GLOBAL VARIABLES CHANGED: None                          *
*    GLOBAL TABLES USED: None                                *
*    GLOBAL TABLES CHANGED: None                             *
*    FILES READ: None                                        *
*    FILES WRITTEN: None                                     *
*    MODULES CALLED: check_range_of_dates,                   *
*                    get_day_time,                           *
*                    build_head,                             *
*                    build_consumption_report                *
*    CALLING MODULES: main menu                              *
*    AUTHOR: Maj. Adilson Marques da Cunha                   *
*            (BRAZILIAN AIR FORCE)                           *
*    HISTORY: None                                           *
*************************************************************/

report_generator ()
  {
        int report_num, fly_id, today, now, date [2],
            aircraft [16], num_ac, error;
        printf ("Please input report number and flying unit
number.  ");
        scanf ("%d %d", &report_num, &fly_id);
        printf ("Please input start and end dates (YYMMDD).
");
        scanf ("%d %d", &date [0], &date [1]);
        error = check_range_of_dates (date);
        if (error) return;
        get_day_time (&today, &now);
        error = build_head (report_num, fly_id, today, now,
                date, aircraft, &num_ac);
        if (error) return;
        switch (report_num)
        {
case 7:         build_consumption_report (today, now, date,
                    aircraft, num_ac);
                break;
default:        printf ("Report number %d not implemented
yet.", report_num);
                break;
        }
        return;
  }
```

```
/***********************************************************
*                                                          *
*    DATE: 26 Nov 84                                       *
*    VERSION: 1.0                                          *
*    NAME: build_head                                      *
*    MODULE NUMBER: 1.1.1                                  *
*    FUNCTION: This module builds the report header and    *
*              gets all the aircraft number in the flying  *
*              unit.                                       *
*    INPUTS: report_num, fly_id, today, now, date          *
*    OUTPUTS: aircraft, num_ac                             *
*    GLOBAL VARIABLES USED: None                           *
*    GLOBAL VARIABLES CHANGED: None                        *
*    GLOBAL TABLES USED: None                              *
*    GLOBAL TABLES CHANGED: None                           *
*    FILES READED: None                                    *
*    FILES WRITTEN: None                                   *
*    MODULES CALLED: print_head                            *
*    CALLING MODULES: report_generator                     *
*                                                          *
*    AUTHOR: Maj. Adilson Marques da Cunha                 *
*            (BRAZILIAN AIR FORCE)                         *
*    HISTORY: None                                         *
*                                                          *
***********************************************************/

int build_head (report_num, fly_id, today, now, date,
aircraft, num_ac)
  int report_num, fly_id, today, now, date [], aircraft [],
*num_ac;
  {

##  char report_name [80];
##  int air_craft, rport, fly;
        int i;
        rport = report_num;
        fly = fly_id;

/* checks for legal report number & gets report name */

##      range of r is reportid
##      retrieve (report_name = r.r_name)
##      where r.r_no = rport
        if (report_name [0] == NULLC)
        {
            printf ("Invalid report number %d.",
report_num);
            return ERROR;
        }
        i = 0;
```

```
/* gets all aircraft number in the flying unit */

##      range of f is flyunit
##      retrieve (air_craft = f.a_no)
##      where (f.f_unitno = fly)
##      {
            aircraft [i] = air_craft;
            i = i + 1;
##      }
        *num_ac = i;
        if (i == 0)   /* if no planes it is bad flying unit
*/
        {
            printf ("Invalid flying unit number %d",
fly_id);
            return ERROR;
        }
        print_head (report_num, fly_id, today, now, date,
report_name);
        return 0;
  }
```

```
/*************************************************************
*                                                            *
*    DATE: 26 Nov 84                                         *
*    VERSION: 1.0                                            *
*    NAME: print_head                                        *
*    MODULE NUMBER: 1.1.1.1                                  *
*    FUNCTION: This module prints the report generator       *
*              header                                        *
*    INPUTS: report_num, fly_id, today, now, date,           *
*            report_name                                     *
*    OUTPUTS: None                                           *
*    GLOBAL VARIABLES USED: None                             *
*    GLOBAL VARIABLES CHANGED: None                          *
*    GLOBAL TABLES USED: None                                *
*    GLOBAL TABLES CHANGED: None                             *
*    FILES READED: None                                      *
*    FILES WRITTEN: None                                     *
*    TERMINAL OUTPUT: Report Header                          *
*    MODULES CALLED: None                                    *
*    CALLING MODULES: build_head                             *
*                                                            *
*    AUTHOR: Maj. Adilson Marques da Cunha                   *
*            (BRAZILIAN AIR FORCE)                           *
*    HISTORY: None                                           *
*                                                            *
*************************************************************/

print_head (report_num, fly_id, today, now, date,
report_name)
  int report_num, fly_id, today, now, date [];
  char report_name [];
  {
        printf (
"\f*********************************************
**************************************");
        printf ( "\n\n\nBRAZILIAN AIR FORCE \t\t\t\t\t
R_DATE :
                %d", today);
        printf ( "\nFLYING UNIT OPERATIONAL CONTROL
SYSTEM\t\t\t
                R_TIME : %d", now);
        printf ( "\nFLYING UNIT NO: %10d \t\t\t\t R_SDATE :
%d",
                fly_id, date [0]);
        printf ( "\nREPORT NUMBER:  %8d \t\t\t\t R_EDATE :
%d",
                report_num, date [1]);
        printf ( "\nREPORT NAME: %s", report_name);
        printf (
"\n\n*****************************************
**************************************");
   }
```

```
/*************************************************************
*                                                           *
*    DATE: 26 Nov 84                                        *
*    VERSION: 1.0                                           *
*                                                           *
*    NAME: build_consumption_report                         *
*    MODULE: 1.1.2                                          *
*    FUNCTION: This module actually builds the consumed     *
*              item report main body                        *
*    INPUTS: today, now, date, acft, num_ac                 *
*    OUTPUTS: None                                          *
*    GLOBAL VARIABLES USED: None                            *
*    GLOBAL VARIABLES CHANGED: None                         *
*    GLOBAL TABLES USED: None                               *
*    GLOBAL TABLES CHANGED: None                            *
*    FILES READ: None                                       *
*    FILES WRITTEN: None                                    *
*    MODULES CALLED: sum_sortie_time, sum_item              *
*    CALLING MODULES: report_generator                      *
*                                                           *
*    AUTHOR: Maj. Adilson Marques da Cunha                  *
*            (BRAZILIAN AIR FORCE)                          *
*    HISTORY: None                                          *
*                                                           *
*************************************************************/

build_consumption_report (today, now, date, acft, num_ac)
  int today, now, date [], acft [], num_ac;
  {

##float sort_time;
  float mis_total [MAX_MIS_TYPE];
  int   item_total [MAX_ITEM_TYPE];
  int i, dummy;

##int plane, start_date, end_date, item_qty, re_num, day,
curr_time;
##char miscode [MIS_LEN], dep_loc [LOC_LEN], arr_loc
[LOC_LEN];
##char item_code [ITEM_LEN], air_name [ACFT_LEN];
  char mis_index [MAX_MIS_TYPE] [MIS_LEN];
  char item_index [MAX_ITEM_TYPE] [ITEM_LEN];

        printf ( "\n\nS_MTCODE    A_TYPE         A_NO
S_DEPLOC
S_ARRLOC   S_TIME S_ITCODE      S_ITCOQY\n\n");

        start_date = date [0];
        end_date = date [1];
```

```
        /* init mis_type & item_type to NULL */

        for (i=0; i<MAX_MIS_TYPE; i++)
        {
             mis_index [i] [0] = '\0';
             mis_total [i] = 0;
        }

        for (i=0; i<MAX_ITEM_TYPE; i++)
        {
             item_index [i] [0] = '\0';
             item_total [i] = 0;
        }

        /* gets aircraft type for each aircraft in the unit
*/

        for (i=0;i<num_ac;i++)
        {
            plane = acft [i];
##          range of a is aircraft
##          retrieve (air_name = a.a_type)
##          where (a.a_no = plane)

/* gets all missions flown by that aircraft between
start/end dates */

##          range of m is mistype
##          retrieve (miscode = m.s_mtcode, dep_loc =
m.s_deploc,
##              arr_loc = m.s_arrloc, sort_time = m.s_time,
##              item_code = m.s_itcode, item_qty =
m.s_itcoqy)
##          where (m.r_sdate >= start_date) and (m.r_edate
<= end_date)
##              and (m.a_no = plane)
##          {

                /* prints out missions */

                printf ( "\n%-12s%-12s%8d  %-10s%-10s%4.1f
%-14s%6d",
                miscode, air_name, plane, dep_loc, arr_loc,
sort_time,
                item_code, item_qty);
```

```
                    /* keeps running total of sortie time by
sortie type */

                    sum_sortie_time (miscode, sort_time,
mis_index, mis_total);

                    /* keeps running total of consumed items */

                    sum_items (item_code, item_qty, item_index,
item_total);
##                  }
          }
          printf ( "\n\n*****************************************
*******************************************\n");
          i = 0;
          while (mis_index [i] [0] != '\0')
          {

              /* prints out each mission type & time */

              printf ( "\nMission %13s Total Sortie Time:
%10.2f",
                  &(mis_index [i] [0]), mis_total [i]);
              i = i + 1;
          }
          printf ( "\n\n");

          /* prints out each item & consumption */

          i = 0;
          while (item_index [i] [0] != '\0')
          {
              printf ( "\nTotal %13s Consumed: %10d",
                  &(item_index [i] [0]), item_total [i]);
              i = i + 1;
          }
          printf (
"\n**********************************************
*************************************\n");
          day = today;
          curr_time = now;
          re_num = 7;
/* updates report relation */
##        append to report (r_no = re_num,
##                r_date = day,
##                r_time = curr_time,
##                r_sdate = start_date,
##                r_edate = end_date)
  }
```

```
/**************************************************************
*                                                             *
*    DATE: 26 Nov 84                                          *
*    VERSION: 1.0                                             *
*                                                             *
*    NAME: sum_sortie_time                                    *
*    MODULE: 1.1.2.1                                          *
*    FUNCTION: This module finds match mission (if any)       *
*              and adds sortie times                          *
*    INPUTS: type, sort_time, index, total                    *
*    OUTPUTS: total                                           *
*    GLOBAL VARIABLES USED: None                              *
*    GLOBAL VARIABLES CHANGED: None                           *
*    GLOBAL TABLES USED: None                                 *
*    GLOBAL TABLES CHANGED: None                              *
*    FILES READ: None                                         *
*    FILES WRITTEN: None                                      *
*    MODULES CALLED: find_match                               *
*    CALLING MODULES: build_consumption_report                *
*                                                             *
*    AUTHOR: Maj. Adilson Marques da Cunha                    *
*            (BRAZILIAN AIR FORCE)                            *
*    HISTORY: None                                            *
*                                                             *
**************************************************************/

int sum_sortie_time (type, sort_time, index, total)
  float sort_time, total [];
  char type [], *index;
  {
        int i;

        i = find_match (index, type, MAX_MIS_TYPE, MIS_LEN);
        if (i == ERROR) return i;
        total [i] = total [i] + sort_time;
        return 0;
  }
```

```
/***********************************************************
*                                                          *
*    DATE: 26 Nov 84                                       *
*    VERSION: 1.0                                          *
*                                                          *
*    NAME: sum_items                                       *
*    MODULE NUMBER: 1.1.2.2                                *
*    FUNCTION: This module finds match items (if any)      *
*              and adds items                              *
*    INPUTS: type, item_used, index, total                 *
*    OUTPUTS: total                                        *
*    GLOBAL VARIABLES USED: None                           *
*    GLOBAL VARIABLES CHANGED: None                        *
*    GLOBAL TABLES USED: None                              *
*    GLOBAL TABLES CHANGED: None                           *
*    FILES READ: None                                      *
*    FILES WRITTEN: None                                   *
*    MODULES CALLED: find_match                            *
*    CALLING MODULES: build_consumption_report             *
*                                                          *
*    AUTHOR: Maj. Adilson Marques da Cunha                 *
*            (BRAZILIAN AIR FORCE)                         *
*    HISTORY: None                                         *
*                                                          *
***********************************************************/

int sum_items (type, item_used, index, total)
  int item_used, total [];
  char type [], *index;
  {
        int i;

        i = find_match (index, type, MAX_ITEM_TYPE,
ITEM_LEN);
        if (i == ERROR) return i;
        total [i] = total [i] + item_used;
        return 0;
  }
```

```
/**********************************************************
*                                                         *
*    DATE: 26 Nov 84                                      *
*    VERSION: 1.0                                         *
*    NAME: find_match                                     *
*    MODULE NUMBER: 1.1.2.1.1                             *
*    FUNCTION: This module finds the matching elements    *
*              in an array.                               *
*    INPUTS: index, type, max, array_size                 *
*    OUTPUTS: index                                       *
*    GLOBAL VARIABLES USED: None                          *
*    GLOBAL VARIABLES CHANGED: None                       *
*    GLOBAL TABLES USED: None                             *
*    GLOBAL TABLES CHANGED: None                          *
*    FILES READ: None                                     *
*    FILES WRITTEN: None                                  *
*    MODULES CALLED: sum_sortie_time, sum_items           *
*    CALLING MODULES: None                                *
*    AUTHOR: Maj. Adilson Marques da Cunha                *
*            (BRAZILIAN AIR FORCE)                        *
*    HISTORY: None                                        *
*                                                         *
**********************************************************/

int find_match (index, type, max, array_size)
  char *index, type [];
  int max, array_size;
  {
        int i;
        for (i=0; i<max; i++)
        {
            /* compares index to type subscribed "i", */
            /* if it matches, then */
            /* it returns index id subscribed "i" */
            if (strcmp (index, type) == 0)
                return i;
            /* if it does not match, then */
            /* if type subscribed "i" is NULL, */
            /* it is at the end of a known element, and */
            /* adds index to type list */
            if (*index == '\0')
            {
                strcpy (index, type);
                return i;
            }
            /* else, it looks at next element */

            index = index + array_size;
        }
        return ERROR;
  }
```

```
/*************************************************************
*                                                            *
*    DATE: 26 Nov 84                                         *
*    VERSION: 1.0                                            *
*                                                            *
*    NAME: check_range_of_dates                              *
*    MODULE NUMBER: 1.1.3                                    *
*    FUNCTION: This module checks the start/end dates for    *
*              legal values & range.                         *
*    INPUTS: date                                            *
*    OUTPUTS: Error Flag                                     *
*    GLOBAL VARIABLES USED: None                             *
*    GLOBAL VARIABLES CHANGED: None                          *
*    GLOBAL TABLES USED: None                                *
*    GLOBAL TABLES CHANGED: None                             *
*    FILES READ: None                                        *
*    FILES WRITTEN: None                                     *
*    MODULES CALLED: check_date                              *
*    CALLING MODULES: report_generator                       *
*                                                            *
*    AUTHOR: Maj. Adilson Marques da Cunha                   *
*            (BRAZILIAN AIR FORCE)                           *
*    HISTORY: None                                           *
*                                                            *
*************************************************************/

int check_range_of_dates (date)
  int date [];
  {
    int error;
        if (date [0] > date [1])
        {
            printf ("Error:  End date must be later than the
 start date");
            return ERROR;
        }
        error  = check_date (date [0]);
        if (error)
        {
            printf ("Error:  Invalid starting date.");
            return ERROR;
        }
        error = check_date (date [1]);
        if (error)
        {
            printf ("Error:  Invalid ending date.");
            return ERROR;
        }
        return 0;
  }
```

```
/***********************************************************
*                                                          *
*    DATE: 26 Nov 84                                       *
*    VERSION: 1.0                                          *
*                                                          *
*    NAME: check_date                                      *
*    MODULE NUMBER: 1.1.3.1                                *
*    FUNCTION: This module checks for legal date.          *
*    INPUTS: date                                          *
*    OUTPUTS: Error Flag                                   *
*    GLOBAL VARIABLES USED: None                           *
*    GLOBAL VARIABLES CHANGED: None                        *
*    GLOBAL TABLES USED: NONE                              *
*    GLOBAL TABLES CHANGED: None                           *
*    FILES READ: None                                      *
*    FILES WRITTEN: None                                   *
*    MODULES CALLED: None                                  *
*    CALLING MODULES: check_range_of_dates                 *
*                                                          *
*    AUTHOR: Maj. Adilson Marques da Cunha                 *
*            (BRAZILIAN AIR FORCE)                         *
*    HISTORY: None                                         *
*                                                          *
***********************************************************/

int check_date (date)
  int date;
  {
    int day, month, num_day [13];

        num_day [1] = 31;        num_day [2] = 28;
num_day [3] = 31;
        num_day [4] = 30;        num_day [5] = 31;
num_day [6] = 30;
        num_day [7] = 31;        num_day [8] = 31;
num_day [9] = 30;
        num_day [10] = 31;       num_day [11] = 30;
num_day [12] = 31;

        day = date % 100;
        month = date/100;
        month = month % 100;

        if (month < 1 || month > 12) return ERROR;
        if (day < 1 || day > num_day [month]) return ERROR;
        return 0;
  }
```

```
/***********************************************************
*                                                          *
*     DATE: 26 Nov 84                                      *
*     VERSION: 1.0                                         *
*                                                          *
*     NAME: get_day_time                                   *
*     MODULE: 1.1.4                                        *
*     FUNCTION: This module gets the system's date & time.*
*     INPUTS: None                                         *
*     OUTPUTS: today, now                                  *
*     GLOBAL VARIABLES USED: None                          *
*     GLOBAL VARIABLES CHANGED: None                       *
*     GLOBAL TABLES USED: None                             *
*     GLOBAL TABLES CHANGED: None                          *
*     FILES READ: None                                     *
*     FILES WRITTEN: None                                  *
*     MODULES CALLED: None                                 *
*     CALLING MODULES: report_generator                    *
*                                                          *
*     AUTHOR: Maj. Adilson Marques da Cunha                *
*             (BRAZILIAN AIR FORCE)                        *
*     HISTORY: None                                        *
*                                                          *
***********************************************************/

#include <sys/types.h>
#include <sys/timeb.h>
#include <time.h>

get_day_time (today, now)
  int *today, *now;
  {
        long *tloc, tm, time ();
        struct tm *local, *localtime();

        tloc = &(tm);
        tm = time (tloc);
        local = localtime (tloc);
        *today = local -> tm_year * 10000;
        *today = *today + local -> tm_mon * 100;
        *today = *today + local -> tm_mday;

        *now = local -> tm_hour * 100;
        *now = *now + local -> tm_min;
  }
```

Appendix G

The Third Implementation Level

(Query Retrievals from Database)

Database query retrieval represents the third level of implementation and one of the most efficient ways to retrieve data from a system. Taking advantage of the relational database structure, the following five queries were sucessfully specified, implemented, and retrieved using the INGRES DBMS database manipulation language (DML) specified in references (15) and (16).

1) The First Query

(a) Query Specification.

```
 -------------------------------
| "Which AIRCRAFT TYPE have     |
| consumed the item MISS77 ?"   |
 -------------------------------
```

(b) Query Implementation.

```
* range of a is aircraft
* range of m is mistype
* retrieve (a.a_type) where
*           a.a_no = m.a_no and
*           m.s_itcode = "miss77"
*  g
```

(c) Query Retrieval.

```
   a_type
 ------------
|  mirage    |
 ------------
  (1 tuple)
```

2) <u>The Second Query</u>

(a) Query Specification.

```
 ------------------------------------------------
|  "What CREW MEMBERS                             |
|  are able to perform INTERCEPTION missions ?"  |
 ------------------------------------------------
```

(b) Query Implementation.

```
* range of c is crew
* range of f is flyunit
* range of m is mistype
* retrieve (c.c_ssn, c.c_lname) where
*           c.c_ssn = f.c_ssn and
*           f.a_no = m.a_no and
*           m.s_mtcode = "intercept"
*  g
```

(c) Query Retrieval.

c_ssn	c_lname
391801970	john
591811724	david

(2 tuples)

3) The First Advanced Query

(a) Query Specification.

```
--------------------------------------------------------
| "Which AIRCRAFT from which BAF FLYING UNITS           |
| are available to perform FORMATION missions          |
| on January 14, 1985 at 10 o'clock ?"                 |
--------------------------------------------------------
```

(b) Query Implementation.

```
* range of v is acftaval
* range of f is flyunit
* range of t is mistype
* retrieve (f.f_unitno, f.a_no) where
*            f.a_no = v.a_no and
*            v.a_avdate = 850114 and
*            v.a_avtime = 100000 and
*            v.a_no = t.a_no and
*            t.s_mtcode = "formation"
*  g
```

(c) Query Retrieval.

f_unitno	f.a_no
03110234	2223

(1 tuple)

4) The Second Advanced Query

(a) Query Specification.

> "What qualified CREW MEMBERS,
> from which BAF FLYING UNITS,
> are able to perform NOW,
> a COMBAT mission ?"

(b) Query Implementation.

```
* range of c is crew
* range of f is flyunit
* range of m is mistype
* retrieve (c.c_ssn, c.c_lname) where
*            c.c_fqcode = "1p" or
*            c.c_fqcode = "in" or
*            c.c_fqcode = "1f" or
*            c.c_fqcode = "if" and
*            c.c_ssn = f.c_ssn and
*            f.a_no = m.a_no and
*            m.s_mtcode = "combat"
*  g
```

(c) Query Retrieval.

c_ssn	c_lname
591811724	david

(1 tuple)

5) <u>The Third Advanced Query</u>

(a) Query Specification.

```
------------------------------------------------------
| "What qualified CREW MEMBERS and which AIRCRAFT,   |
|  from which FLYING UNITS,                          |
|  can be employed NOW (i.e., at 0630 Jan 03, 1985), |
|  to perform FORMATION mission in REGION d ?"       |
------------------------------------------------------
```

(b) Query Implementation.

```
* range of c is crew
* range of f is flunit
* range of m is mistype
* range of v is acftaval
* retrieve (f.f_unitno, c.c_ssn,
*           c.c_lname, v.a_no) where
*           c.c_fqcode = "lp" or
*           c.c_fqcode = "in" or
*           c.c_fqcode = "1f" or
*           c.c_fqcode = "if" and
*           c.c_ssn = f.c_ssn and
*           f.f_unitrg = "d" and
*           f.a_no = v.a_no and
*           v.a_avdate = 850103 and
*           v.a_avtime = 063000 and
*           v.a_avsitu = "available" and
*           v.a_no = m.a_no and
*           m.s_mtcode = "formation"
*  g
```

(c) Query Retrieval.

f_unitno	c_snn	c_lname	a_no
03110234	691801613	Foster	2301

(1 tuple)

The implementation of these advanced query transactions in this thesis constitute only a sample of what could be done in order to improve the efficiency levels of the flying unit operational control system environment.

Bibliography

1. Atre, S. Data Base: Structure Techiniques for Design, Performance, and Management . New York: Wiley-Interscience Publication, 1980.

2. Barr, Avron and others. The Handbook of Artificial Intelligence (First Edition). New Jersey: The Library of Computer and Information Sciences, 1981.

3. Cabral, Paulo Dantas "The Philosophy of Computation in the Brazilian Department of the Air Force," Brazilian Air Force Staff Command School Project , 1975.

4. Chamberlin, Donald D. "A History and Evaluation of System R" Communications of the ACM , 24 : 632-646 (October 1981).

5. Clark, Thomas D. Jr. "Decision Making in Organizations," Lecture Paper on SM 6.48 Management Systems Analysis and Simulation II Course . School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, (May 1984).

6. Clark, Thomas D. Jr. "Decision Support Systems," Lecture Paper on SM 6.48 Management Systems Analysis and Simulation II Couse . School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, (June 1984).

7. Codd, E. F. "A Relational Model of Data for Large Shared Data Banks," Communications of the ACM , 13 : 377-387 (June 1970).

8. Cross, Capt Stephen E. "Requirements for Thesis Research," AFIT Advisor's Thesis Handout , 1 : 1-3 (February 1984).

9. Cunha, Maj Adilson M. da "Automatic Data Processing System for Flight Statistics," Brazilia Data Processing Center System Manual , 1 : 12-32 and 2 : 8-14 (February 1983).

10. Cunha, Maj Adilson M. da Management System Analysis and Simulation Project . School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, June 1984.

11. Cunha, Maj Adilson M. da "The Design and Implementation of an Automatic Data Processing System

for Flight Statistics in the Brazilian Air Force," Brazilian Air Force School of Improvement Officers Magazine , 16 : 19-22 (August 1980).

12. Date, C. J. An Introduction to Database Systems (Third Edition). Readings: Addison-Wesley Publishing Company, 1982.

13. Davis, Richard M. Thesis Projects in Science and Engineering (First Edition). New York : St. Martin's Press Inc., 1980.

14. Department of the U.S. Air Force. Style Guide for Theses and Dissertations. Wright-Patterson Air Force Base: Air Force Institute of technology, 1983

15. Epstein, Robert A Tutorial on INGRES . Berkeley: Electronics Research Laboratory, College of Engineering, University of California, 1977.

16. Epstein, Robert Creating and Maintaining a Database Using INGRES . Berkeley: Electronics Research Laboratory, College of Engineering, University of California, 1977.

17. Hendrix, Gary G. "Natural-Language Processing, The Field in Perspective," BYTE Magazine , 99 : 304-352 (September 1981).

18. Hirsch, Abraham "Artificial Intelligence Comes of Age," Computer & Electronics Magazine , 22 : 63-96 (March 1984).

19. Kermighan, Brian W. The C Programming Language . New Jersey: Prentice-Hall, 1978.

20. Martin, James An Information Systems Manifesto . New Jersey: Prentice-Hall Inc., 1984.

21. Martin, James Managing the Data-Base Environment New Jersey: Prentice-Hall Inc., 1983.

22. Nilsson, Nils J. Principles of Artificial Intelligence (First Edition). Palo Alto: Tioga Publishing Company, 1980.

23. Pereira Filho, Jorge da Cunha "Banco de Dados Hoje" Dados e Ideias - Brazilian Magazine , 99 : 55-63 (February 1979).

24. Rich, Elaine "The Gradual Expansion of Artificial Intelligence," Computer Magazine , 99 : 4-12 (May

1984).

25. Silva, Victorio B. da "The Informatic and the Development of Its Applications in the Brazilian Air Force," Brazilian Air Force Staff Command School Project, 203 : 7-8 (December 1982).

26. Sprague, Ralph H. Jr. and Eric D. Carlson. Building Effective Decision Support Systems. New Jersey: Englewood Cliffs, 1982.

27. Townsend, C. Using dBase II. Berkeley: Osborne & MCGraw-Hill, 1984.

28. Waltz, David L. "An English Language Question Answering System for a Large Relational Database," Communications of the Association for Computing Machinery, 21 : 526-539 (July 1980).

29. Waltz, David L. "Natural Language Access to a Large Data Base," Naval Research Reviews (January 1976).

30. Weiner, James L. "The Logical Record Keeper: Prolog on the IBM," BYTE Guide to the IBM PC, 9 : 125-130 (September 1984).

31. White, Capt Roger P. A Natural Language Interface for a Prolog Database. MS Thesis GCS/EE/83D-22. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, December 1983.

32. Winograd, Terry "GUS, A Frame-Driven Dialog System - The Language Understander Project at the Xerox Palo Alto Research Center", Artificial Intelligence Magazine, 8 : 157-173 (May 1976).

33. Winston, Patrick Henry Artificial Intelligence (Second Edition). Readings: Addison-Wesley Publishing Company, 1984.

34. Woodfill, John and others. INGRES Reference Manual (Version 6.3). Berkeley: Electronics Research Laboratory, College of Engineering, University of California, 1981.

V I T A

Major Adilson Marques da Cunha was born on 12 October 1948 in Belem, Para, Brazil. He graduated in 1970 with a Bachelor of Science (B.S.) degree from Academia da Forca Aerea Brasileira (Brazilian Air Force Academy) and in 1978 with a B.S. degree in Business Administration from the Centro de Ensino Unificado de Brasilia (United Educational Center of Brazilia), Brazil. He completed pilot training and received his wings in January 1970.

As an officer and pilot in the Brazilian Air Force (BAF), he has worked at the operational level since 1970, accumulating nearly four thousand flying hours, in both training and operational missions. As he progressed in rank, his responsibilities slowly shifted more and more to managerial and control activities. In 1975, as head of a Squadron and Group Flight Statistics Section, he began to develop a computerized file management system for flight statistics and decision making. He worked in nearly all phases of the system, from the problem definition to the final test and implementation phase, until entering the Master's degree course at the Air Force Institute of Technology School of Engineering, in June 1983.

Permanent address: Rua Monsenhor Magaldi 50 / Apt 201
Jardim Guanabara - Ilha do Governador
Rio de Janeiro - CEP 21940 - BRAZIL

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION
Unclassified

1b. RESTRICTIVE MARKINGS

2a. SECURITY CLASSIFICATION AUTHORITY

2b. DECLASSIFICATION/DOWNGRADING SCHEDULE

3. DISTRIBUTION/AVAILABILITY OF REPORT
Approved for public release; distribution unlimited

4. PERFORMING ORGANIZATION REPORT NUMBER(S)
AFIT/GCS/ENG/84D-7

5. MONITORING ORGANIZATION REPORT NUMBER(S)

6a. NAME OF PERFORMING ORGANIZATION
School of Engineering

6b. OFFICE SYMBOL *(If applicable)*
AFIT/ENG

7a. NAME OF MONITORING ORGANIZATION

6c. ADDRESS *(City, State and ZIP Code)*
Air Force Institute of Technology
Wright-Patterson AFB, Ohio 45433

7b. ADDRESS *(City, State and ZIP Code)*

8a. NAME OF FUNDING/SPONSORING ORGANIZATION
Brazilian Air Force

8b. OFFICE SYMBOL *(If applicable)*
BAF

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER

8c. ADDRESS *(City, State and ZIP Code)*
Brazil, Brasilia, D.F.

10. SOURCE OF FUNDING NOS.

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT NO.

11. TITLE *(Include Security Classification)*
See Box 19

12. PERSONAL AUTHOR(S)
Adilison Marques da Cunha, B.S., Major, BAF

13a. TYPE OF REPORT
MS Thesis

13b. TIME COVERED
FROM ________ TO ________

14. DATE OF REPORT *(Yr., Mo., Day)*
1984 December 14

15. PAGE COUNT
244

16. SUPPLEMENTARY NOTATION

17. COSATI CODES

FIELD	GROUP	SUB. GR.

18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
Relational Databases
Decision Support Systems
Database Query Retrieval
Artificial Intelligence
Natural Language System

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

Title: A DataBASE Design For The Brazilian Air Force Flying Unit Operational Control System

Thesis Chairman: Stephen E. Cross, Captain, USAF

Approved for public release: IAW AFR 190-17
LYNN E. WOLAVER
Dean for Research and Professional Development
Air Force Institute of Technology (ATC)
Wright-Patterson AFB OH 45433

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT
UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS ☐

21. ABSTRACT SECURITY CLASSIFICATION
Unclassified

22a. NAME OF RESPONSIBLE INDIVIDUAL
Stephen E. Cross, Captain, USAF

22b. TELEPHONE NUMBER *(Include Area Code)*

22c. OFFICE SYMBOL

DD FORM 1473, 83 APR

EDITION OF 1 JAN 73 IS OBSOLETE.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

Abstract

This thesis addresses a relational database design for a Brazilian Air Force Flying Unit Operational Control System. After defining the problem and specifying requirements, an overall system analysis was performed using Decision Support System Theory. A top-down planning for decision support systems and databases, and a functional analysis were performed to identify potential environmental database applications. Using a canonical approach, a file management system was mapped to a database conceptual model and afterwards to a database logical model.

A prototype Dialog Generator Management Software was implemented through menu driven programs, using the dBASE II DBMS version 2.1 running on a Z-80 microcomputer. The database partial implementation was performed using the INGRES DBMS version 7.10 running on a VAX 11/780, from which advanced queries were retrieved. Finally, an investigation of optimum query retrievals from databases is performed using Artificial Intelligence methods and techniques.

END

FILMED

4-85

DTIC